Institute for Software Technology
Graz University of Technology
Graz

Intelligent Techniques for Software Release Planning

Philipp Ghirardini, BSc
philipp.ghirardini@student.tugraz.at

August 2010

Supervision:
Felfernig, Alexander, Univ.-Prof. Dipl.-Ing. Dr.techn.

Abstract

Requirements engineering is a crucial part of every system engineering and software development process. Requirements reflect the needs of the stakeholders and therefore a project's success can be measured on how well they are met. Furthermore, nowadays software is not released when all requirements are completed, but in small, incremental releases. This prioritization of the requirements has a great impact on the stakeholders satisfaction. In this master thesis a release planner is proposed that is implemented on the basis of constraint technologies. The work identifies constraints concerning releases, required and available resources, and relations between requirements. It is shown that release plans can be generated without developing individual scheduling algorithms, but using generic ones provided by the Java Constraint Programming library. Furthermore, the integration of an existing diagnosis framework is presented, that helps to identify inconsistencies in a requirements model. The generation of alternative release plans makes it necessary to sort them referring to preferences of the stakeholders. Therefore, a ranking approach is introduced, based on the Multi Attribute Decision Theory. For demonstration purposes a web-based prototype is developed that provides a modelling environment and the possibility to create and rank release plans as well as diagnosing inconsistencies.

Keywords: Constraint Programming, Constraint Satisfaction Problems, Requirement Engineering, Release Planning, Model-based Diagnosis, Multi Attribute Decision Theory

Zusammenfassung

Die Anforderungsanalyse, besser bekannt unter Requirements Engineering ist eine entscheidende Phase in jedem Projektplanungs- und jedem Softwareentwicklungsprozess. Die in dieser Phase erhobenen Anforderungen sollen die Bedürfnisse und Wünsche aller Projektteilnehmer möglichst gut wiederspiegeln. Die Zufriedenheit der Stakeholder kann daher in weiterer Folge als Maßstab für den Erfolg eines Projekts herangezogen werden. Heutzutage werden Software Projekte nicht mehr als ein großes, fertiges Produkt veröffentlicht, sondern in kleinen, aufeinander aufbauenden Versionen, sogenannten Releases. Daraus ergibt sich die Notwendigkeit, die einzelnen Anforderungen zu priorisieren und zu Releases zusammenzufassen. In der folgenden Diplomarbeit wird dieses Problem mit Hilfe von Constraints Technologien gelöst. Ein wesentlicher Bestandteil jedes Constraint Satisfaction Problems is die Identifikation der Constraints. Im Bereich der Release Planung müssen vorhandene, sowie benötigte Ressourcen in Betracht gezogen werden, aber auch die Beziehungen zwischen Anforderungen haben starken Einfluss auf die Planung. Im Falle von Widersprüchen in einem erstellten Anforderungsmodel wird ein Diagnose Framework integriert, welches die Ursachen von Inkonsistenzen identifiziert. Die Erzeugung einer Vielzahl von unterschiedlichen Plänen erfordert eine vernünftige Sortierung der gefundenen Lösungen. Zu diesem Zweck wird ein Reihungsverfahren, basierend auf der Multi Attribute Decision Theory vorgestellt, welches die Pläne auf Grund von Bewertungen der einzelnen Anforderungen reiht. Zu Demonstrationszwecken wurde ein webbasierter Prototyp entwickelt, mit dessen Hilfe Anforderung modelliert, Pläne erzeugt und Inkonsistenzen gefunden werden können.

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am ……. ……..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

……. ……..
date (signature)

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible.

First of all I would like to thank my supervising tutor Univ.-Prof. Dipl.-Ing. Dr.techn. Alexander Felfernig for his abundant help and his prolific suggestions.
I would also like to gratefully acknowledge the support of Dipl.-Ing. Monika Schubert and Dipl.-Ing. Monika Mandl.

I dedicate this thesis to my parents who unremittingly support me during my years of study. They made this work possible.

I would like to thank all of my friends, especially Dr. Michael Taschner for prove-reading of the English manuscript.

Finally, I express my profound appreciation to my girlfriend Ina for her encouragement, understanding, and patience during this thesis.

Contents

Chapter 1

Introduction

1.1 Background

An important part of software development projects is the requirements engineering process. In this early stage of a project the functions a system shall provide are determined. This requires a detailed analysis of the software's application domain. Requirements engineering is not only used for software projects, but also for almost every development and engineering process. Therefore, it is a well researched field. In Section 3 the basic concepts and techniques of requirements engineering are presented. It is shown that the elicitation of requirements depends on stakeholders and the success of a project can be measured by their achievements. In addition to the problem of selecting the right requirements, their order of implementation is crucial for a project's success. This prioritization process is called release planning. It considers the stakeholders' preferences, available and required resources and tries to schedule the requirements within a given number of releases. This topic is presented in detail in Section 3.3. In this master thesis release planning is implemented on the basis of constraints technologies. This software paradigm is not new, and especially widely-used in the area of configuration systems and scheduling problems. However, in recent years it is also more and more frequently used in other areas. The differences between constraint programming and conventional programming are described in Chapter 2, furthermore the underlying ideas and advantages are presented there. Chapter 4 proposes a way how requirements, their resources, and their interdependencies can be modelled as a constraint satisfaction problem and how release plans can be generated based on this model.

1.2 Motivation And Objectives

Later in this master thesis the importance of requirements engineering and especially of release planning is emphasized. It is also shown that applications that support this

processes need to be very flexible and easy to use because of changing requirements and the different background of the stakeholders. In other areas, like configuration systems, constraint programming is already successfully used to approve highly flexible applications. The objective of this master thesis is to show that constraint technologies are applicable for generating alternative release plans based on a given requirements model. Such models include a number of requirements, their needed resources, and their interdependencies among themselves. At the end of the work a prototype shall be presented that allows to model requirements, resources, and releases in a way that release plans can be created by constraint solvers. Therefore, a detailed analysis of requirements engineering and release planning has to be done (see Section 3). Furthermore, it is necessary to understand how a constraint satisfaction problem can be modelled and how solutions can be generated. It should be determined what aspects are essential for a release plan and what constraints can be identified in the requirements engineering process. This does not only mean to identify resources that are needed by a requirement, but also to find relations between requirements and how they can be modelled. This model shall be implemented with a Java constraint programming library. Furthermore, an existing diagnosis framework [Felfernig et al., 2009] shall be integrated and used to identify inconsistencies in a given requirements model. Finally an example application shall be analyzed and modelled with the implemented prototype. This use case shall show the functionality of the application as much as identify advantages and problems of the proposed method.

1.3 Results

The following work shows that constraint programming can be used for release planning. It presents how releases, required, and available resources, and dependencies between requirements can be modelled by using the Java Constraint Programming Library (JaCoP)[1]. For demonstration purposes, a web-based prototype is developed that provides an environment for modelling requirements and their relations. Based on a created model the presented constraint solver generates a number of alternative release plans. Furthermore, the created plans are sorted by their overall stakeholders' satisfaction. Therefore, the Multi Attribute Decision Theory (MAUT)[Felfernig et al., 2006] is used to calculate a rating for each release plan. In order to identify inconsistencies in a generated model a diagnosis framework [Felfernig et al., 2009] is integrated. Since the presented diagnosis tool locates the constraints that can not be fulfilled, the user of the prototype can easily adjust the model.

[1] http://www.osolpro.com/jacoptwiki/bin/view.pl/Main/WebHome

Chapter 2

Constraint Programming

2.1 Overview

Since the last few years constraint programming can be found in different fields of applications. CP is used for classical configuration problems (e.g. telecommunication network [Barták, 1999]), scheduling and planning [Van Beek and Chen, 1999], recommendation (e.g. financial services [Felfernig et al., 2006]), analysis (e.g. DNA structure analysis [Barták, 1999]) or for test case generation (e.g. model-based testing [Pretschner et al., 2001]). Because of its potential, especially in the area of modelling heterogeneous optimization and satisfaction problems it has been identified as

> "... one of the strategic directions in computer research."

by the ACM (Association for Computing Machinery) [Barták, 1999; Marriott and Stuckey, 1998].
However, constraint programming is not a new software paradigm. Already in 1963 Sutherland [1963] presented the Sketchpad system that was a constraint language for graphical interaction. Also other early constraint programming languages like Ref-Arf [Fikes, 1970], Lauriere's Alice [Lauriere, 1978], Sussmann's CONSTRAINTS [Sussman et al., 1980], and Borning's ThingLab [Borning, 1981] supported the three key issues of Constraint Programming. In accordance with Wallace [1996] these are:

- declarative problem modelling and efficient constraint enforcement,
- propagation of the effects of decisions,
- flexible and intelligent search for feasible solutions.

Today a lot of imperative programming languages offer CP libraries, including the most popular ones like Java (Choco[1], JaCop[2]), C++ (Disolver[3], Gecode[4]), and Python (python-

[1] http://www.emn.fr/z-info/choco-solver/
[2] http://jacop.osolpro.com/
[3] http://research.microsoft.com/apps/pubs/default.aspx?id=64335
[4] http://www.gecode.org/

constraint[5]). Some solutions even provide interfaces to different programming languages, like IBM's ILOG CPLEX CP Optimizer[6].
But what does constraint programming actually mean and what are the differences to conventional programming? One definition was given by Saraswat et al. [1991]:

> "The store-as-valuation conception of von Neumann computing is replaced by the notion that the store is a constraint (a finite representation of a possibly infinite set of valuations) which provides partial information about the possible values that variables can take."

Gallaire [1985] and J. Jaffar [1987] commented that logic programming (LP) is a particular kind of constraint programming. They further emphasize that LP and in general declarative programming does not instruct *how* a problem has to be solved but *what* problem has to be solved. That is very close to the idea of constraint programming. The literature distinguishes two branches of CP. The most common one is *constraint satisfaction*. This type of CP works with finite domains. The type used less frequently is called *constraint solving*. It does not distinguish how a problem is described but instead of finite domains, infinite or more complex domains are used. This master thesis deals with the first branch, constraint satisfaction problems (CSP) . The next subsections gives an introduction to constraint programming (CP). In Subsection 2.2 the basic terms and definitions of CP are introduced. Three small examples are presented in Subsection 2.3 to show the use of constraint programming. The used search algorithms and their improvements are covered in Subsection 2.4 and finally basic types of constraints and some global constraints are presented.

2.2 Basic Definitions

2.2.1 Constraint Satisfaction Problem

A CSP is defined as a tuple (X, D, C):

- a set of *variables* $X = x_1, x_2, \ldots, x_n$,
- for each variable x_i, a finite set D_i of possible values (its domain), and
- a set of *constraints* $C = c_1, c_2, \ldots, c_m$ restricting the values that the variables can simultaneously take.[Barták, 1999]

[5]http://www.eveutilities.com/products/emma
[6]http://www-01.ibm.com/software/integration/optimization/cp-optimizer/

Russell et al. [1995] define CSP in a quite similar way. Furthermore, they define the state of a problem as an *assignment* of values to some or all of the variables. This definition is important for understanding *solutions* to a CSP.

2.2.2 Solutions

Barták [1999] defines a *solution* as an assignment of a value's domain to every variable, in such a way that all constraints are satisfied. Three types of desired solutions can be distinguished:

- just one solution, with no preference as to which one,
- all solutions,
- an optimal solution that maximizes a given objective function.

How a solution can be found and what aspects have to be considered regarding CSP is presented in Section 2.4.1. The next section provides some examples for a better understanding of constraints satisfaction problems.

2.3 Examples

This section presents three frequently used CSP examples. It should help to understand how a constraint problem can be modelled, how a solution looks like, and how miscellaneous CP can be used. The first example is a map colouring problem, the second one shows the solution of a cryptarithmetic puzzle, and the last one deals with a scheduling problem. The first two examples are taken from Russell et al. [1995] and the third one from Marriott and Stuckey [1998].

2.3.1 Map Colouring Problem

The objective of this example is to color each territory of Australia either red, green or blue. But no neighbours should have the same color. Figure 2.1 (a) shows seven regions of Australia. Figure 2.1 (b) represents the problem of (a) as a constraint graph.

In this examples the variables are *WA, NT, Q, NSW, V, SA* and *T*, which present the seven states of Australia. The domain of each variable is set to *red, green, blue*. This means that every variable can be assigned to red or green or blue.

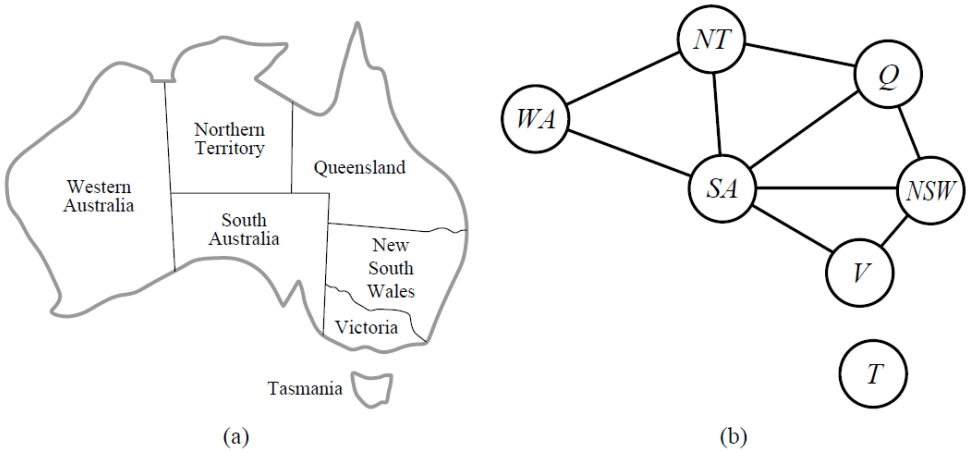


Figure 2.1: (a) The principal states and territories of Australia. (b) Representation as a constraint graph [Russell et al., 1995].

Based on the constraint graph of Figure 2.1 (b) the following constraints are required to solve the problem:

- $WA \neq NT \wedge WA \neq SA$
- $NT \neq SA \wedge NT \neq Q$
- $SA \neq Q \wedge SA \neq NSW \wedge SA \neq V$
- $Q \neq NSW$
- $NSW \neq V$

After having defined the variables, the domains, and the constraints, as described in 2.2.1 a solution can be searched. One possible solution is displayed in Figure 2.2. The presented solution has the following variable assignments $\{$ *WA=red; NT =green; Q=red; NSW =green; V=red; SA=blue; T=red*$\}$. This is only one of multiple solutions. As mentioned in Subsection 2.2.2 most solvers provide the possibility to search for all solutions or optimize an objective function. So it would be possible to search a solution where the usage of the color red is minimized.

2.3.2 Cryptarithmetic Puzzle

Another frequently used example to illustrate the usage and possibilities of constraint programming are cryptarithmetic puzzles. This kind of mathematical games consist of a mathematical equation among unknown numbers. Every number is represented by a different character. The objective of this game is to find a value for each character. Some examples are shown in Figure 2.3. Hence in example (a) digits for the characters *T, W,*

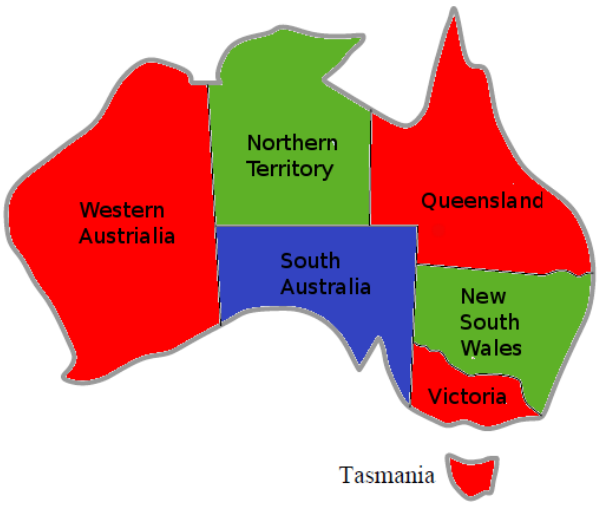


Figure 2.2: Possible solution of the map colouring problem in the case of Australia.

T W O	S E N D
+ T W O	+ M O R E
F O U R	M O N E Y
(a)	(b)

Figure 2.3: Two cryptarithmetic puzzles.

O, F, U, R have to be found. Each character can be replaced by a digit from 0 to 9. This means that we have the variables $\{T,W,O,F,U,R\}$ and each variable's domain is set to $\{0 \dots 9\}$. One constraint says that all characters have to present a different digit. This could be modelled as in the map-colouring example above $T \neq W \wedge T \neq O \dots$. Fortunately, a global constraint is available for such cases, the *Alldiff* constraint. A more detailed explanation of global constraints is given in Section 2.4.7. *Alldiff* (T, W, O, F, U, R) guarantees that in a valid solution all variables have different values assigned. Finally, the constraints for the addition have to be defined. In such examples also the carry bit has to be taken into account. Therefore the auxiliary variables X_1, X_2, X_3 are needed to model the carry bit (0 or 1) for all positions.

$O + O = R + 10 * X_1$

$X_1 + W + W = U + 10 * X_2$

$X_2 + T + T = O + 10 * X_3$

$X_3 = F$

One solution for this problem is

$$T => 4, W => 6, O => 9, F => 0, U => 3, R => 8.$$

Example (b) can be modelled in the same way and a valid solution looks as follows:

$$S => 9, E => 5, N => 6, D => 7, M => 1, O => 0, R => 8, Y => 2.$$

Besides multiple other examples the JaCoP library[7] also provides executable implementations of these puzzles.

2.3.3 Scheduling Problem

The last example illustrates an easy way to solve a small scheduling problem. The objective of this example is to move four pieces of furniture. Four people are available for the move and it has to be finished within one hour. Table 2.1 presents the duration for each furniture item and the required number of people. The variables S_p, S_c, S_b, S_t represent the start

Item	Time required to move	No. of people required
piano	30	3
chair	10	1
bed	15	3
table	15	2

Table 2.1: Resource table for furniture moving.

times for moving the piano, the chair, the bed, and the table. Because of the required durations of each furniture piece the following domains can be determined

$$S_p = [0 \dots 30],\ S_c = [0 \dots 50],\ S_b = [0 \dots 45],\ S_t = [0 \dots 45].$$

Since the piano requires 30 minutes to be moved and 60 minutes are available for all four furniture items, the start time has to be within the first 30 minutes $S_p = [0 \dots 30]$. Starting to move the piano later than 30 minutes would make it impossible to complete the tasks within 60 minutes. As in the example above CP provides a global constraint for this problem, it is called *Cumulative*. This example can be modelled as follows:

"**Cumulative**($[S_p, S_c, S_b, S_t]$, $[30, 10, 15, 15]$, $[3, 1, 3, 2]$,4)".

This constraint takes the start time variables as first parameter. The second argument represents the durations it takes to move the furniture items, and the third one takes the required resources. So the piano S_p requires 30 minutes and 3 people to be moved, whereas the chair S_c only needs 10 minutes and 1 person. The last argument defines the

[7] http://jacop.osolpro.com/

total number of available resources, in this case it represents the 4 movers. Based on this information the problem can be solved. Figure 2.4 presents the problem graphically, assuming that the piano is scheduled at the beginning. Now the chair, the bed, and the table have to be placed into the empty space. The fact that the piano shall be scheduled first means to assign the value 0 to variable S_p. Since the bed and the table cannot be moved at the same time as the piano the domains of S_b and S_t can be reduced to:

$$S_b = [30 \dots 45],\ S_t = [30 \dots 45].$$

Next the chair variable S_c can be assigned to 0, because it only needs one person to move. S_b can be assigned to 30 and S_t to 45 or vice versa. A more detailed explanation about reducing variables' domains by constraint propagation and global constraints is given in the Subsections 2.4.5 and 2.4.7. However, it should be mentioned that this constraint can find inconsistencies very quickly. Supposing that only a total time of 50 minutes is available, the constraint identifies that the minimum required time has to be 60 minutes, because the piano, bed, and table cannot be moved at the same time.
Marriott and Stuckey [1998] describe this example in detail. The book also describes the power and detailed functionality of the *Cumulative* constraint. An executable implementation of this example is provided on the JaCoP website[8].

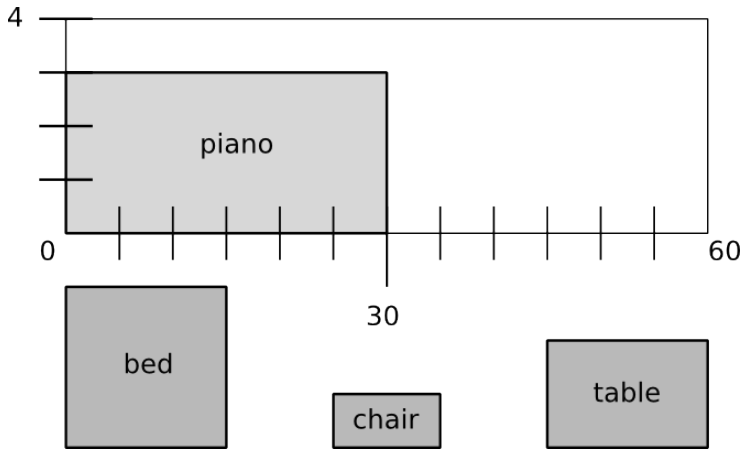


Figure 2.4: Moving four furniture items within 60 minutes and with four people. Scheduling is done with the help of the *Cumulative* constraint.

[8]http://jacop.osolpro.com/

2.4 Searching

This section gives a short overview about searching methods in CP. The formulation of CSP as a search problem allows to use a lot of search algorithms to solve a constraint satisfaction problem. Common algorithms like breadth-first search (BFS) can be used to solve satisfaction problems. But using a BFS will result in a tree with $n! * d^n$ leaves, when d values can be assigned to any of n variables, even though there are only d^n possible complete assignments [Russell et al., 1995]. Fortunately, all CSPs are *commutative*. This means that the order of application of any given set of actions has no effect on the outcome. Therefore, all CSP search algorithms

> "... generate successors by considering possible assignments for only a single variable at each node in the search tree."[Russell et al., 1995].

Hence the number of leaves can be reduced to d^n which is equal to the number of possible complete solutions. The next subsections describe the most important search techniques of CP.

2.4.1 Backtracking

Backtracking belongs to the family of complete algorithms. Unlike incomplete algorithms it guarantees that a solution will be found if one exists. Therefore, complete algorithms are able to determine an optimal solution and the nonexistence of a solution [Rossi et al., 2006a]. A backtracking search is used for a depth-first search. It traverses the search tree recursively starting at the root node. At each node c it is checked if a solution is still possible. If so the algorithm continues at the next child of c. If a valid solution is not possible a backtrack to c's parent is done and the whole sub-tree can be skipped. A backtracking algorithm for constraint satisfaction problems is shown as pseudo code in Algorithm 1. Unfortunately this plain algorithm does not perform very well. In other domains the performance can be increased by adding domain-specific heuristic functions. However, CSP algorithms should be generic and work independently of its application area. Barták [1999] notes three major disadvantages using a standard backtracking algorithm for CSP.

1. Trashing, i.e.: repeated failure due to the same reason.
2. Redundant work, i.e.: conflicting values of variables are not remembered.
3. Late detection of the conflict, i.e.: conflict is not detected before it really occurs.

Algorithm 1 A simple backtracking algorithm for CSP [Russell et al., 1995]

```
function RECURSIVE-BACKTRACKING(assignment,csp) returns
a solution or failure
  if assignment is complete then
    return assignment
  end if
  var ← SELECT-UNASSIGNED-VARIABLE(VARIABLES

                                   csp

  ,assignment,csp)
  for all value in ORDER-DOMAIN-VALUES(var,assignment,csp) do
    if value is consistent with assignment according to CONSTRAINTS[csp] then
      add {var = value} to assignment
      result ← RECURSIVE-BACKTRACKING(assignment, csp)
      if result ≠ failure then
        return result
      end if
      remove {var = value} from assignment
    end if
  end for
  return failure
```

Russell et al. [1995] also mentioned that an efficient general-purpose backtracking algorithm has to deal with the following questions:

1. Which variable should be assigned next and in what order should its values be tried?
2. What are the implications of the current variable assignments for the other unassigned variables?
3. When a path fails, can the search avoid repeating this failure in subsequent paths?

The next few sections present some methods which are solving or at least soften the noted drawbacks and answer these questions.

2.4.2 Variable And Value Ordering

A relevant aspect for high-performance backtracking algorithms is the question which variable should be taken next. As in CSP no domain-knowledge should be used to answer this question, other methods are necessary.
One of them is called *minimum remaining values* (MRV) heuristic or *minimum remaining values*. MRV selects the variable with the fewest legal values or in other words the variable which is most likely to fail. Finding an illegal state early in the search tree allows pruning

the tree and avoids pointless searching. Benchmarking by Russell et al. [1995] showed that MRV can improve the performance by factor 3 to 3000. Another heuristic called *degree heuristic* can be used to select the first variable or when MRV does not provide a clear decision. This method chooses the variable which is involved in the largest number of constraints on other unassigned variables.
After a variable is selected the order of values can influence the performance. This is only true if just one solution is asked, it does not matter if all solutions are requested. One method is called *least-constraining-value* heuristic. It tries to keep as much flexibility as possible for subsequent variable assignments, i.e. it selects the value which eliminates the lowest possible number of values of the neighbouring variables.

2.4.3 Backjumping And Backmarking

Backjumping (BJ) is a method to avoid trashing. BJ is working identically to backtracking but differs when a backtrack has to be done. While simple backtracking returns to the immediate past variable, backjumping goes to the *most recent* conflicting variable. This variable can be found by identifying a *conflict set* based on the violated constraint. Russell et al. [1995] define a conflict set as:

> "The conflict set for variable X is the set of previously assigned variables that are connected to X by constraints."

Another problem of BT is redundant work. *Backchecking* (BC) and *backmarking* (BM) can help to avoid this drawback. Both methods reduce the number of compatibility checks by remembering incompatible assignments. This means that if an assignment $X \leftarrow b$ is incompatible to another assignment $Y \leftarrow a$ then $Y \leftarrow a$ is not considered as long as $X \leftarrow b$. Beside redundant constraint checking BM also avoids redundant discoveries of inconsistencies by remembering all incompatible labels for every label. Furthermore, it reduces compatibility checks by remembering already succeeded ones [Barták, 1999].

2.4.4 Forward Checking

The presented look back methods have the drawback of late conflict detection. Other approaches are *look ahead* strategies which detect inconsistency before it occurs. A simple example is forward checking (FC). If a variable X is assigned FC checks all variables Y, that are connected to X by a constraint and removes all values of Y's domain, which are inconsistent to the labeled variable X. Russell et al. [1995] also mentioned that

> "... every branch pruned by backjumping is also pruned by forward checking.".

2.4.5 Local Consistency

Although forward checking can find a lot of inconsistencies it has restrictions because it does not look far enough. A further approach in finding inconsistencies are consistency techniques. Cooper and Schiex [2004] describe local consistency as follows:

- Local consistency is a relaxation of consistency, which means that for any consistent CSP there is an equivalent non-empty locally consistent CSP.
- This equivalent locally consistent CSP, which is unique, can be found in polynomial time by so-called enforcing or filtering algorithms.

This means that local consistency condition can be strengthened by a transformation. The so called *constraint propagation* is changing the problem but not the solution. This can be done by reducing the domains of variables or strengthening constraints. The presented search algorithms are based on the observation that if a domain of any variable becomes empty, then the CSP is unsatisfiable. Consistency techniques transform a CSP into an equivalent CSP in which the variables' domains are reduced. An equivalent solution means

> "... that the constraints represented by the CSPs have the same set of solutions."[Marriott and Stuckey, 1998]

If a domain of any variable in the transformed CSP becomes empty, then also the original CSP is unsatisfiable. Combining consistency based solvers with backtracking improves the performance of the search algorithm by reducing the variables' domain at each step in the search. The following subsections present the two most knowing local consistency conditions: *Node Consistency* and *Arc Consistency*.

Node Consistency

Node Consistency (NC) is the simplest consistency technique. This technique removes all values from a variable's domain that are inconsistent with unary constraints [Barták, 1999]. This means that only values are remaining which satisfy the constraint on this domain. An example could be the constraint $X \leq 2$ and X's domain is set to $0, 1, 2, 3, 4$. NC would remove 3 and 4 and reduces the domain to $0, 1, 2$.

Arc Consistency

Arc Consistency (AC) is the most widely used consistency technique. It removes values from variables' domains which are inconsistent with binary constraints. A problem is arc consistent if every variable is arc consistent with any other one. The following example

shall illustrate the functionality of AC. Consider the simple constraint $X < Y$ and both variables' domains are set to $1, 2, 3$. This means X can never be 3 and Y can never be 1. As a consequence, value 1 can be removed from Y's domain and value 3 from X's domain. This example is illustrated in Figure 2.5.

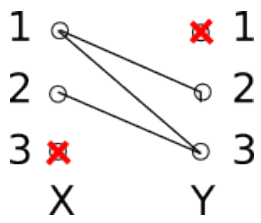


Figure 2.5: Arc Consistency.

Several AC algorithms exist. (AC-1 to AC-7). All of them are based on repeated arcs until a domain is empty or a consistent state is reached. AC has to be done repeatedly until no more inconsistency exists, because removing values of a variable's domain can produce inconsistency at variables that are pointed in that arc. The most popular are AC-3 and AC-4 [Barták, 1999]. AC can be done before the search process or after every assignment step. The latter algorithm is sometimes called *maintaining arc consistency* (MAC).

2.4.6 Primitive, Logical, And Conditional Constraints

Primitive constraints include basic arithmetic operations ($+$, $-$, $*$, $/$) or basic relations ($=$, $\neq$, $<$, $\leq$, $>$, $\geq$). Arithmetic operations and relations can be modelled between variables or between variables and constants. These types of constraints can be used for logical and conditional constraints. The most important of them are:

- $\neg c$
- $c_1 \Leftrightarrow c_2$
- $c_1 \wedge c_2 \wedge \ldots \wedge c_n$
- $c_1 \vee c_2 \vee \ldots \vee c_n$.

Some libraries also provide conditional constraints like:

- if c_1 then c_2
- if c_1 then c_2 else c_3.

2.4.7 Global Constraints

Van Hoeve and Katriel [Rossi et al., 2006b] are defining a global constraint as follows:

> "A global constraint is a constraint that captures a relation between a non-fixed number of variables."

It should be mentioned that all global constraints can be expressed as a conjunction of several simpler constraints that were described before. However, there are two good reasons for introducing global constraints. One of them concerns the simplification of programming. The cryptarithemtic puzzle in Section 2.3.2 uses the global constraint *Alldiff* instead of modelling all distinctions between the characters. The same applies to the global constraint *Cumulative* in the scheduling example 2.3.3. But beside the ease of programming, global constraints contribute extensively to good performance in CP. As shown in Section 2.4.5 constraint propagation is a proper technique to reduce the search space and to increase performance. Global constraints are working in a similar way. This type of propagation is called *filtering*. This task considers all not assigned variables and removes all inconsistent values from its domain. Determining if a value is useful for the CSP is NP-hard. Therefore, filtering is done separately with respect to each constraint. If a value is useless, regarding to one constraint, it is also useless for the whole CSP, but not vice versa. Good constraints remove many values with less computational costs. Two important global constraints (*Alldifferent, Cumulative*) have already been presented in Section 2.3. Some other global constraints which are also used later in this work (see Chapter 4) are:

- *Sum* The sum of a list of variables $x_1, x_2, \ldots, x_n$ is equal to a variable y.
- *Min* The minimum of a list of variables $x_1, x_2, \ldots, x_n$ is equal to a variable y.
- *Max* The maximum of a list of variables $x_1, x_2, \ldots, x_n$ is equal to a variable y.
- *Diff2* can be used to schedule two-dimensional rectangles.

  > "The *Diff2* constraint takes as an argument a list of two-dimensional rectangles and assures that for each pair of $i, j (i \neq j)$ of two-dimensional rectangles, there exists at least one dimension k where i is after j or j is after i." [Kuchcinski and Wolinski, 2003]

A detailed explanation of these constraints and further global constraints like *Element, Knapsack, Cardinality or Circuit Constraint* can be found in Chapter 6 of Rossi et al. [2006b].

Chapter 3

Requirements Engineering

3.1 Overview

This chapter introduces the requirements engineering process. It emphasizes on the importance of RE in the system engineering process, explains the main iterations, and presents the role of release planning. Before going into more detail, a definition of RE is given:

> "Requirements engineering is the branch of software engineering concerned with the real-world goals for functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families."[Zave, 1997]

In other words, it means to identify the requirements of a system, validating if they are what the stakeholders want, and specifying what the developers have to build. But it also means to verify the delivered work and to maintain changing specifications. Even though RE takes place at the beginning of the system engineering process, it influences the entire development. Figure 3.1 shows the main parts of a generic system engineering process.

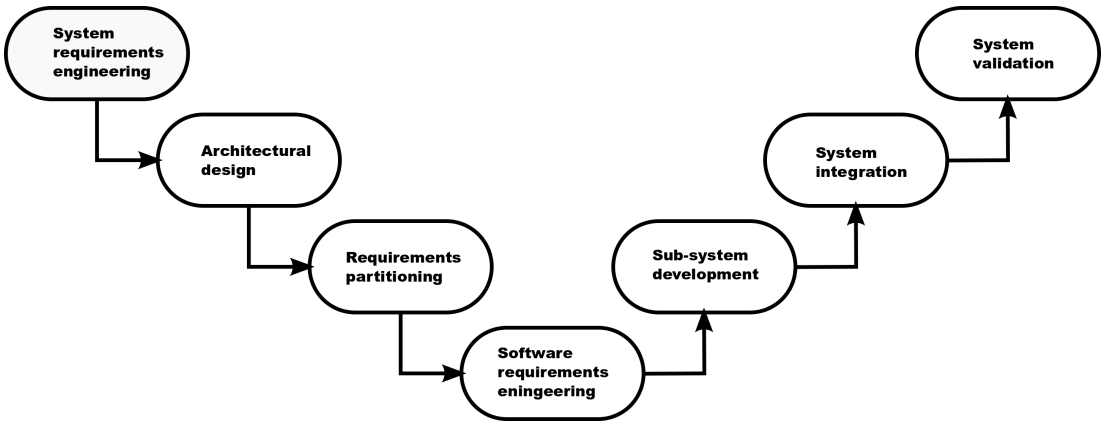


Figure 3.1: The system engineering process [Sommerville and Kotonya, 1998].

Activity	Description
System requirements	The requirements for the system as a whole are established. These will usually be expressed in a fairly high-level fashion and written in natural language. Some detailed constraints (such as compatibility constraints) may be included if these are critical for the success of the system.
Architectural design	The system is decomposed into a set of independent sub-systems.
Requirements partitioning	The requirements are partitioned to these sub-systems. At this stage, decisions may be made about whether requirements should be hardware or software requirements.
Software requirements engineering	The high-level software requirements are decomposed into a more detailed set of requirements for the software components of the system.
Sub-system development	The hardware and the software subsystems are designed and implemented in parallel.
System integration	The hardware and software subsystems are put together to complete the system.
System validation	The system is validated against its requirements.

Table 3.1: Detailed description of a system engineering process.[Sommerville and Kotonya, 1998]

A detailed description of the seven steps of Figure 3.1 is given in Table 3.1. The following work focuses on the fourth step of Figure 3.1, "Software requirements engineering". As mentioned in Table 3.1, requirements have to be found in an early state of the engineering process and are the basis for validating the completed system. Thus a lot of research and publications regarding different aspects of RE in different areas already exist. This chapter gives a short overview of the main aspects of RE, especially related to the software development process. The next sections introduce the process of requirements engineering, with a presentation of the key issues and some further supporting techniques. The last section of this chapter is focused on release planning in software development. It presents the core activities of release planning and approaches to support this process.

3.2 Requirements Engineering Process

This section presents the basic terms and definitions of RE and introduces the process of requirements engineering. The main purpose of RE is to identify and formalize the needs and expectations of different stakeholders of a software project. Thus RE can be seen as a process to consolidate these expectations and needs of the stakeholders. Nuseibeh and

Easterbrook [2000] identify five core activities of RE to reach this purpose:

1. eliciting requirements,
2. modelling and analysing requirements,
3. communicating requirements,
4. agreeing requirements and
5. evolving requirements.

Sommerville [1996] illustrates the requirements analysis process as a cyclic process of seven activities (see Figure 3.2). The next subsections show that the core activities identified by Nuseibeh and Easterbrook [2000] are quite well covered by the process of Figure 3.2. Before

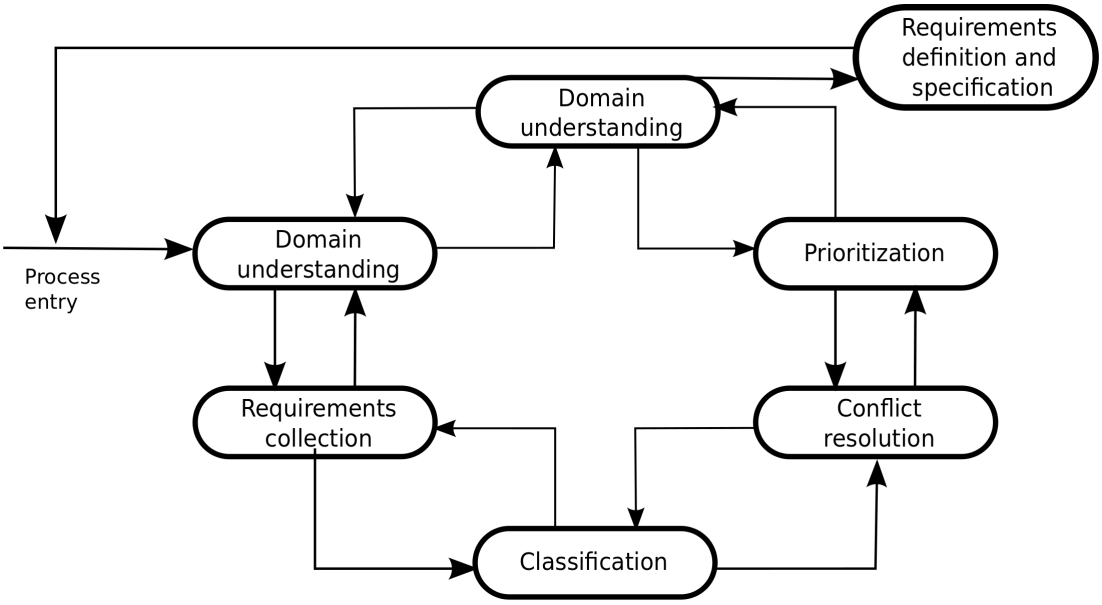


Figure 3.2: The requirements analysis process [Sommerville, 1996].

describing these core elements in more detail, some frequently used terms will be defined. Sommerville [1996] identified the following terms:

- A *requirements definition* is a statement in a natural language of what services the system is expected to provide. It also defines the environment in which it has to operate. The information is supplied by the customers.
- A *requirements specification* is a structured document describing the system services in detail. This document should be very precise and it is often used as a contract between developers and customers.
- A *software specification* is a further refinement of the requirements specification and is used as the basis for software design and implementation.

- A *stakeholder* in RE is a person that has either direct or indirect influence on the system requirements.
- *Functional requirements* define the services that a system has to provide, how it should react to a particular input and its behaviour in specific situations. However, a functional requirement can also define what a system should not do in a particular situation.
- *Non-functional requirements* are defining constraints of services or functions. These are for example timing, performance or security constraints. However, also constraints regarding the development process or standards belong to non-functional requirements.

3.2.1 Eliciting Requirements

Eliciting requirements is the first part of the RE-process. Before any other steps can be carried out, the requirements have to be collected. Unfortunately it is not enough to simple ask the users what they want. Eliciting requirements needs a careful analysis of the environment in which the system will be used. This includes an analysis of the organisation, the application domain, and the business processes [Sommerville and Kotonya, 1998]. An important step in the RE-process is to identify the stakeholders of the project. Since the success of a project depends on the contentedness of the people involved in it, they should be integrated early in the process. In software development projects these are often the project management, the development team, and the sales or marketing department on the one hand, and on the other hand the customers and the end-users, whose satisfaction is crucial for the success of the project [Nuseibeh and Easterbrook, 2000]. The end-users have to be distinguished into novice, expert, and occasional users. Sommerville and Kotonya [1998] identify four dimensions that have to be understood when eliciting requirements:

1. *Application domain understanding*
   The development of a system for a specific domain needs sufficient knowledge and background information about this domain.
2. *Problem understanding*
   In addition to the general knowledge of a domain, further information about detailed processes and needs of a particular domain is necessary. It helps to extend general domain knowledge.
3. *Business understanding*
   Normally, systems are influencing how an organisation or business is working. For

reaching a real gain it is necessary to understand in detail how the different parts of the organisation work and how they interact.

4. *Understanding the needs and constraints of system stakeholders*
   As mentioned above, stakeholders are all people who are more or less influenced by the system. It is necessary to understand their specific demands on a supporting system. Therefore, it is essential to analyze the process that a system shall support, as well as consider already existing systems.

Considering all four dimensions is essential for the success of a software project, because a system will only be accepted if it meets the customers/users needs and supports their activities. However, it is not easy to analyze the four mentioned dimensions. Some problems of this analysis are listed by Sommerville and Kotonya [1998]. Understanding an application domain can be very difficult, because specialized literature often uses domain specific terminology. In some very specific areas such literature does not even exist and the developers depend on experts. Understanding a particular problem in an organisation also depends on the cooperation of experts, who are often too busy or not interested in supporting the RE-process. In some companies competitive departments try to influence the requirements to their own advantages. Finally, some stakeholders are not able to articulate their requirements or even do not know what they really want from a new system. Furthermore, requirements tend to change over the development process, either because of wrong definitions, misunderstandings between the stakeholders or changing circumstances. Hence there is not one solution that works in all cases. Some elicitation techniques are proposed by Nuseibeh and Easterbrook [2000]:

1. *Traditional techniques* include questionnaires, surveys, interviews, and analysing existing documentation, process models or established standards.
2. *Group elicitation techniques* means to promote stakeholders to participate early in the process. Brainstorming and workshops are used to elicit requirements [Ncube and Maiden, 1999].
3. *Prototyping* can support eliciting requirements when stakeholders have difficulties to find requirements or to support other techniques. Prototypes have to be developed early in the development process and are often discarded before the actual product development begins (throw-away prototyping) [Sommerville and Kotonya, 1998].
4. *Model-driven techniques* provide a model of how information shall be captured. Models can be divided into goal-based models like KAOS [Van Lamsweerde et al., 1998] or scenario-based models like CREWS [Maiden, 1998].
5. *Cognitive techniques* are techniques of knowledge acquisition and shall help to ex-

ploit requirements by analysing stakeholders while performing a particular task. One example is *protocol analysis*, where an expert is observed while he is performing a special process. During this activity he has to think loud.

This subsection shall show that eliciting requirements is more difficult than it seems. A wide range of stakeholders have to be taken into account and it is not easy to capture the requirements which meet the needs of all people involved in the project. Although some techniques were developed to support this process, the selection of the right one depends heavily on the application area.

3.2.2 Modelling And Analysing Requirements

Nuseibeh and Easterbrook [2000] describes modelling as

> "the construction of abstract descriptions that are amenable to interpretation".

The focus is on 'abstract descriptions", so it does not mean to create an alternative representation of a system, but to simplify it and to pick out the most significant characteristics [Sommerville, 1996]. *Data-flow models*, *semantic data models*, and *object models* are the widely used ones to analyze requirements [Sommerville and Kotonya, 1998; Sommerville, 1996]. Data-flow models are concentrating on representing how data flows through a sequence of processing steps. Each processing step identifies a potential function and thus a requirement. Furthermore, data stores can be found by a visualization of a data-flow. Advantages of such models are that data-flow diagrams are easy to explain and can also be created by non-developers.
Semantic data models do not represent the system process, but try to show the relations between data. Almost every software system makes use of large databases or other information. By associating this data and their attributes, requirements can be found and analyzed. The representation of this model is often done with entity-relation diagrams (ER-diagrams) that are also widely used in database design. Theses types of diagrams are more and more replaced by UML.
Object models represent the real world with the help of objects and relations among them. While an object itself is representing a real-world entity, relations are described by connections. As known from UML-diagrams, objects consist of a name, attributes, and services. These models also often use aggregations and inheritances for illustrations.
There are many approaches dealing with requirements modelling and analysing. Some of them are considered in modelling the real world and derive requirements from these models, others focus on the analysis of the requirement [Wieringa, 1996].

3.2.3 Communicating Requirements

As mentioned above, requirements affect many different stakeholders with different background knowledge and different terminology. Furthermore, requirements and stakeholders can change over time. Hence it is important to formalize requirements in a way that they can be understood by the stakeholders without or with only little knowledge. A variety of formal, semi-formal, and informal languages already exist for this purpose [Nuseibeh and Easterbrook, 2000].
Another important part of communication is *requirement traceability*(RT) . A definition of RT is given by Gotel and Finkelstein [1994]:

> "Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through periods of on-going refinement and iteration in any of these phases)".

This is especially important for the requirements' management to estimate impacts of changes and to analyze the re-process.

3.2.4 Agreeing Requirements

After the requirements are elicited and analyzed they have to be agreed by the stakeholders. In the RE-process this step is also often called *requirements validation*. Sommerville and Kotonya [1998] describe requirements validation as

> '... check the requirements to certify that they present an acceptable description of the system which is to be implemented."

Therefore, two questions have to be answered in this stage of the requirements engineering process:

- *Validation*: Have we defined the right requirements?
- *Verification*: Have we defined the requirements right?

At the end of this process a requirement document that includes standardized and detailed descriptions of all requirements shall exist. This document has to be agreed by the stakeholders and is used as contract between the supplier and the customer of a system. Therefore, the abovementioned questions have to be answered with "yes" by the stakeholders. Sommerville [1996] proposes to check the following aspects during this process:

- *Validity:* A user may think that a system is needed to perform a certain function.

However, further thoughts and analysis may identify additional or different functions that are required. Systems have diverse users with different needs and any set of requirements is inevitably a compromise across the user community.

- *Consistency:* No requirement should conflict with any other.
- *Completeness:* The definition should include all functions and constraints intended by the system user.
- *Realism:* There is no point in specifying requirements that are unrealizable. It may be acceptable to anticipate some hardware developments but developments in software technology are much less predictable.

There are two big problems in validating requirements. The first problem concerns the question if it can be proven that the right requirements were elicited. Assuming that

> "... requirements describe some objective problem that exists in the real world, and that validation is the task of making sufficient empirical observations to check that this problem has been captured correctly" [Nuseibeh and Easterbrook, 2000]

the answer has to be no. The reason for this conclusion is given by Popper, who supposes that scientific theories can never be proven correct through observations, they can only be refuted [Popper, 2002]. Another problem are disagreements among the different stakeholders. Since stakeholders have different interests in the system, conflicting goals have to be resolved during the validation process. One approach to solve this problem is given by Boehm et al. [1995], where they propose to identify the most important goals of the individual stakeholders and meet these ones instead of all. Some methods exist to support this process. Four frequently used methods are:

- requirements reviews,
- prototyping,
- model validation, and
- requirements testing.[Sommerville and Kotonya, 1998]

Requirements validation and verification is often disregarded or underestimated in projects, even though it provides the basis for design and implementation. Errors in the final requirements can produce up to 100 times higher costs than programming errors, because requirement changes imply design and implementation changes [Sommerville, 1996].

Requirement type	Description
Mutable requirements	Requirements that change because of changes to the environment in which the organization is operating.
Emergent requirements	Requirements that emerge as the customer's understanding of the system develops during the system development. The design process may reveal new emergent requirements.
Consequential requirements	Requirements that result from the introduction of the computer system. Introducing the computer system may change the organization's processes and open up new ways of working which generate new system requirements.
Compatibility requirements	Requirements that depend on the particular systems or business processes within an organization. If these change, the compatibility requirements on the commissioned or delivered system may also have to evolve.

Table 3.2: Volatile requirements. [Sommerville, 1996].

3.2.5 Evolving Requirements

Large software projects can take several years of development time. Hence requirements can change during this time for different reasons. During long development periods the business objectives can change and these have to be reflected in the requirements. Furthermore, missing or wrong requirements during the initialization phase necessitate adaptions of the requirements. Another reason could be that the stakeholders' needs are better understood during the development phase [Sommerville, 1996]. However, adding or removing requirements may also be necessary. The main reasons for deleting requirements are to save money or because of time problems [Nuseibeh and Easterbrook, 2000]. Generally, requirements in the evolution phase can be divided into *enduring requirements* and into *volatile requirements*. While the first ones are relatively stable and often reflect the core activities of a domain, the second ones are likely to change over time [Sommerville, 1996]. Table 3.2 presents a further subdivision of volatile requirements that illustrates the high probability of changes in requirements. As already noted in Subsection 3.2.3 changing a requirement often influences others and requires to repeat the validation process. Hence evolving requirements are a major challenge for requirement management.

3.3 Release Planning In Software Development Projects

3.3.1 Overview

The requirements engineering process in Section 3.2 shows the main aspect of finding requirements. In contrast release planning (RP) deals with the problem in which order requirements shall be implemented. In monolithic-type development a system is released after all requirements are implemented and tested, so RP is not necessary. Nowadays this approach is more and more replaced by incremental software development, where the system is deployed as a series of releases. Hence it is more likely that the users support the system and provide feedback in an early state of the system engineering process. Each release provides additive functionality and improvements of the previous release. A definition of RP is given by Amandeep et al. [2003]:

> "Release planning for incremental software development includes the assignment of requirements to releases such that all technical, risk, resource, and budget constraints are fulfilled."

As already mentioned in Section 3.1, a successful software system covers the needs of the stakeholders. In addition to the selection of the right requirements, their prioritization also has a big influence on the overall satisfaction of the stakeholders. In accordance to the definition above and the stakeholders' satisfaction, Ruhe and Saliu [2005] identify four main characteristics of good release plans:

- It provides maximum business value by offering the best possible blend of features in the right sequence of releases.
- It satisfies most important stakeholders involved.
- It is feasible in terms of the existing resource capacities available.
- It reflects existing dependencies between features.

The enumeration above uses the term feature instead of requirement. Features are abstractions of requirements and can be defined as

> "... a logical unit of behavior that is specified by a set of functional and quality requirements."[Bosch, 2000]

Features are basically used to alleviate the communication between developers and customers as well as managing the size and complexity of requirements [Ruhe, 2005]. E.g.: A requirement might be that users have to authenticate before gain access to a system. Although this description could be sufficient for the users of a system, it is too abstract and vague for the development team. Therefore, this requirement, respectively this feature

is divided into the following sub-requirements:

- Developing a *Login Window* that allows to enter user name and password,
- a *Database Table* to store the user names and passwords,
- *Database Interfaces* for communication with the database,
- *User Management Tool* to create, modify, and delete users,
- and *Password Encryption* for encrypting passwords before they are stored in the database.

A user only has a benefit if all sub-requirements are fulfilled and integrated into a system. This can be called a feature. Hence, users focus more on features and developers are more interested in the individual requirements. The presented prototype of Chapter 4 shows a way to model features by various interdependencies of requirements. The following four subsections present four main aspects of RP [Ruhe and Saliu, 2005].

3.3.2 Dependencies Between Features

A variety of dependencies exist between requirements which are discussed later in Section 4.4.3. This subsection shows three dependencies between features, proposed by Ruhe and Saliu [2005]. The first type defines that a feature Fi has to be implemented before a feature Fj. E.g.: If a user has to authenticate before he can use a system, the *Authentication* feature has to be implemented before all other features. The second one couples features that Fi has to be in the same release as Fj. This is useful if Fi does not work without Fj and vice versa. E.g.: A video sharing platform shall allow to upload video files and share them with other users. One feature provides the possibility to upload files and another allows to watch them. In this case it is necessary to assign these features to the same release, because it does not make sense to upload files when it is not possible to watch them, as much as it is useless to provide a video player when no content is available. Furthermore, he mentions a dependency where features influence the value of another feature. This means that features which are implemented in the same release have a different additive value or effort as if they would be implemented separately [Ruhe, 2005]. E.g.: The video platform shall provide auto-generated metadata. The collection of metadata (date of upload, size, used codec, user, ...) can easily be done during the upload, because it hardly needs more than a database table to store it. If this feature is implemented in a later release than the *Upload* feature, it is difficult or expensive to reconstruct some meta-data from already uploaded files (e.g.: codec). Some metadata, like the user who provided a video can not be recovered at all.

3.3.3 Resource Constraints

An essential part of RP are resource constraints. Every requirement needs different types and amounts of resources, unfortunately these are limited for each release. Hence scheduling requirements has to guarantee not to exceed these limits. Typical resource types are:

- requirements specification,
- analysis and design,
- implementation,
- testing,
- money and
- risk [Ruhe, 2005].

3.3.4 Stakeholders

Like in the whole system engineering process, stakeholders are also a crucial factor in RP. As mentioned in Section 3.1 stakeholders have different approaches to the system and therefore different importance. Ruhe and Saliu [2005] propose to classify stakeholders according to their importance (very-low/low/medium/high/extremely-high). Features that are preferred by important stakeholders shall be released early. This has to be taken into account during the creation of the release plan.

3.3.5 Prioritization Of Features

Ruhe [2005]; Ruhe and Saliu [2005] present the dimensions *value* and *urgency* to evaluate priority. The value shall express the importance of a feature for the final system and shall be considered independent of time. He proposes a nine-point scale to measure a feature's value, where one means a very low value and nine an extremely high value. The second dimension urgency shall reflect the impact on stakeholders' satisfaction regarding to the time a feature is released. He recommends that the stakeholders represent their satisfaction by dividing nine points for each feature to the preferred release or to postpone it. Assuming two releases, a stakeholder s can express his preference that feature f shall be in the first release through $Sat(s, i) = (9, 0, 0)$. $Sat(s, i) = (3, 3, 3)$ means that the time of release does not matter for stakeholder s. If a feature shall be postponed entirely, it can be expressed through $Sat(s, i) = (0, 0, 9)$.

3.3.6 Known Approaches

A well-known approach of release planning is called *planning game*. This type of release planning is often used in agile development, like *extreme programming* (XP). This process is divided into two parts, the release planning and the iteration planning. In XP this is done very often, sometimes once a week. During the release planning phase developers and customers are present. The customers write down their requirements and the developers estimate the effort for each of them. Finally the customers select the most important requirements until the total available effort is reached. While this approach works well in smaller projects it may fail if the complexity of projects increases [Amandeep et al., 2003]. Especially the presence of the customers and their different importance is hard to handle in large projects with this approach.
Another approach is called EVOLVE*. It is

> "...designed as an iterative and evolutionary procedure mediating between the real world problem of software release planning, the available tools of computational intelligence for handling explicit knowledge and crisp data, and the involvement of human intelligence for tackling tacit knowledge and fuzzy data." [Ruhe, 2005]

Every iteration consists of three phases:

1. In the *modelling* phase the real world problems are modelled to make it suitable for computational intelligence based solution techniques. There the constraints, decision variables, and their dependencies are defined.
2. The second step is called *exploration* and is focused on an application that is generating alternative solutions and evaluates it according to certain criteria.
3. *Consolidation* is the last phase of an iteration. Human decision makers evaluate the solutions and refine the underlying model for the next iteration.

The reason for the iterative part of EVOLVE is to allow all kinds of late changes in requirements like changing prioritization, effort estimates or changing weights assigned to stakeholders. This information is used as input to determine the next iteration k+1 [Greer and Ruhe, 2004]. This approach is illustrated in Figure 3.3.

Greer and Ruhe [2004] emphasize that the generated solutions by EVOLVE are optimal or near optimal, but optimality can not be guaranteed. Because of the iterative approach the used genetic algorithms produce different solutions if the algorithms are applied several times. Greer and Ruhe [2004] emphasize several advantages of EVOLVE over existing methods:

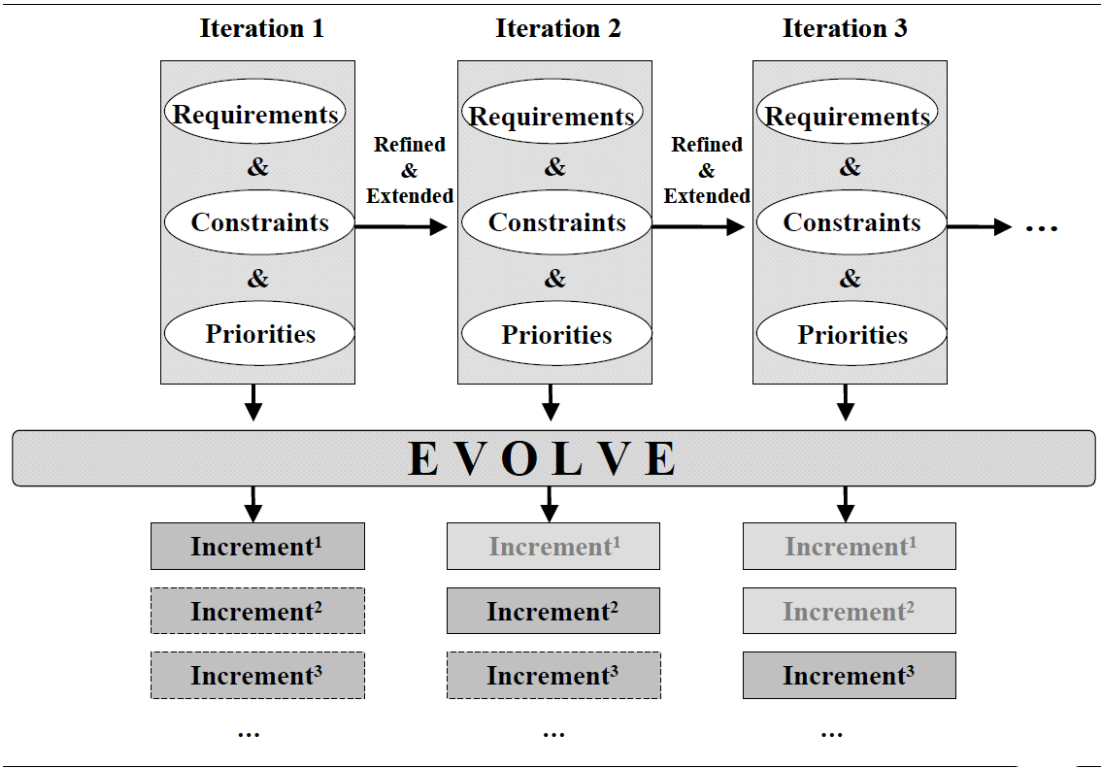


Figure 3.3: EVOLVE approach to assignment of requirements to increments [Greer and Ruhe, 2004].

- EVOLVE takes into account stakeholders priorities as well as effort constraints for all releases.
- EVOLVE assumes that software requirements are delivered in increments.
- EVOLVE considers inherent precedence and coupling constraints.
- It offers greater flexibility by allowing changes in requirements, constraints, and priorities.
- It recognizes conflicting stakeholders priorities.
- A set of promising candidate solutions is generated and one can finally be chosen by the decision maker.

A software based on this approach was developed by the University of Calgary[1]. A release plan created by this application is shown in Figure 3.4. The next chapter presents a different approach for generating release plans. A modelling environment is presented that allows to create a requirements model including the elicited requirements, their required resources, and their interdependencies. In contrast to EVOLVE, the proposed approach

[1]http://www.releaseplanner.com/

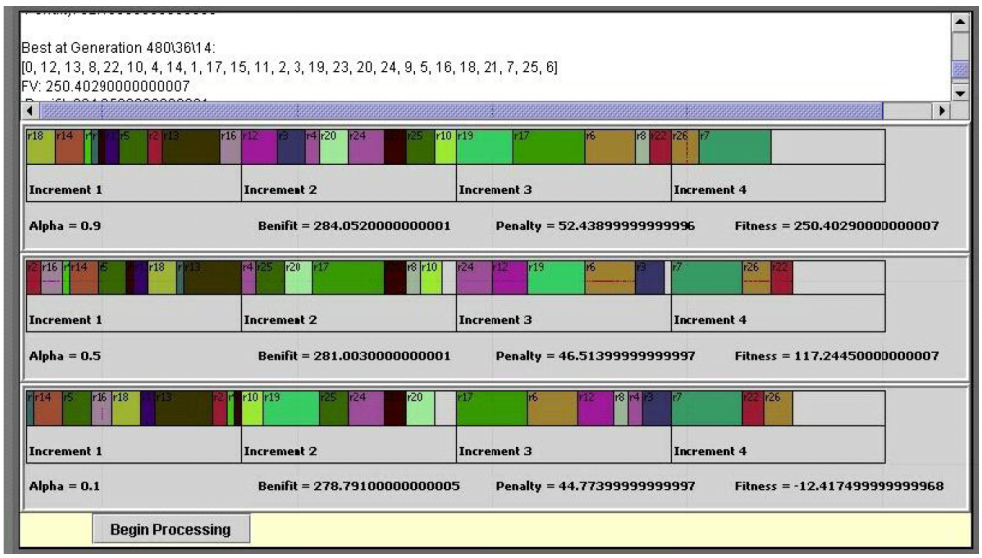


Figure 3.4: Release plan, generated by the EVOLVE approach [Amandeep et al., 2003].

is focused on scheduling single requirements instead of features. However, features can be modelled by defining certain relations between requirements. This allows to create a more detailed requirements model and from there the generation of a more detailed release plan. Another difference is the kind of generating alternative release plans. While EVOLVE uses an individual scheduling algorithm, the proposed approach uses constraints technologies for generating plans. This allows to create a number of alternative plans, but also to search for an optimal solution. It also makes it possible to define release start and end ranges instead of fixed start and end times. Furthermore, an diagnosis component can be used to identify inconsistencies between requirements or the lack of resources.

Chapter 4

Practical Work

4.1 Overview

This chapter shows how release planning can be done with constraint programming technologies. The main objectives of this chapter are:

- Identifying variables and constraints in requirements engineering and release planning.
- Developing a prototype that allows users to create a requirements model.
- Creating release plans with the Java Constraint Programming library JaCoP[1].
- Diagnosing inconsistencies in a requirements model.
- Ranking of the created release plans based on stakeholders' ratings.

The next few sections show how the mentioned objectives can be achieved. First some literature is analyzed to identify the required constraints for release planning. The second section gives an overview of the used technologies for the implemented web-based prototype and presents the basic functions. Section 4.3.1 introduces the core elements of the prototype. It is shown how the identified constraints and variables can be modelled and how a solution looks like. Afterwards a diagnosis component is presented, that allows debugging and finding of inconsistencies in the requirements model. The last section shows how the created plans can be ranked by their overall stakeholders' satisfaction.

4.2 Prototype Requirements Analysis

Before developing a user interface and modelling the release planner the domain has to be understood. This section identifies the most significant elements of requirements engineering and release planning. On the one hand this section detects information that has to be provided by the user of the prototype and on the other hand the various interdependencies

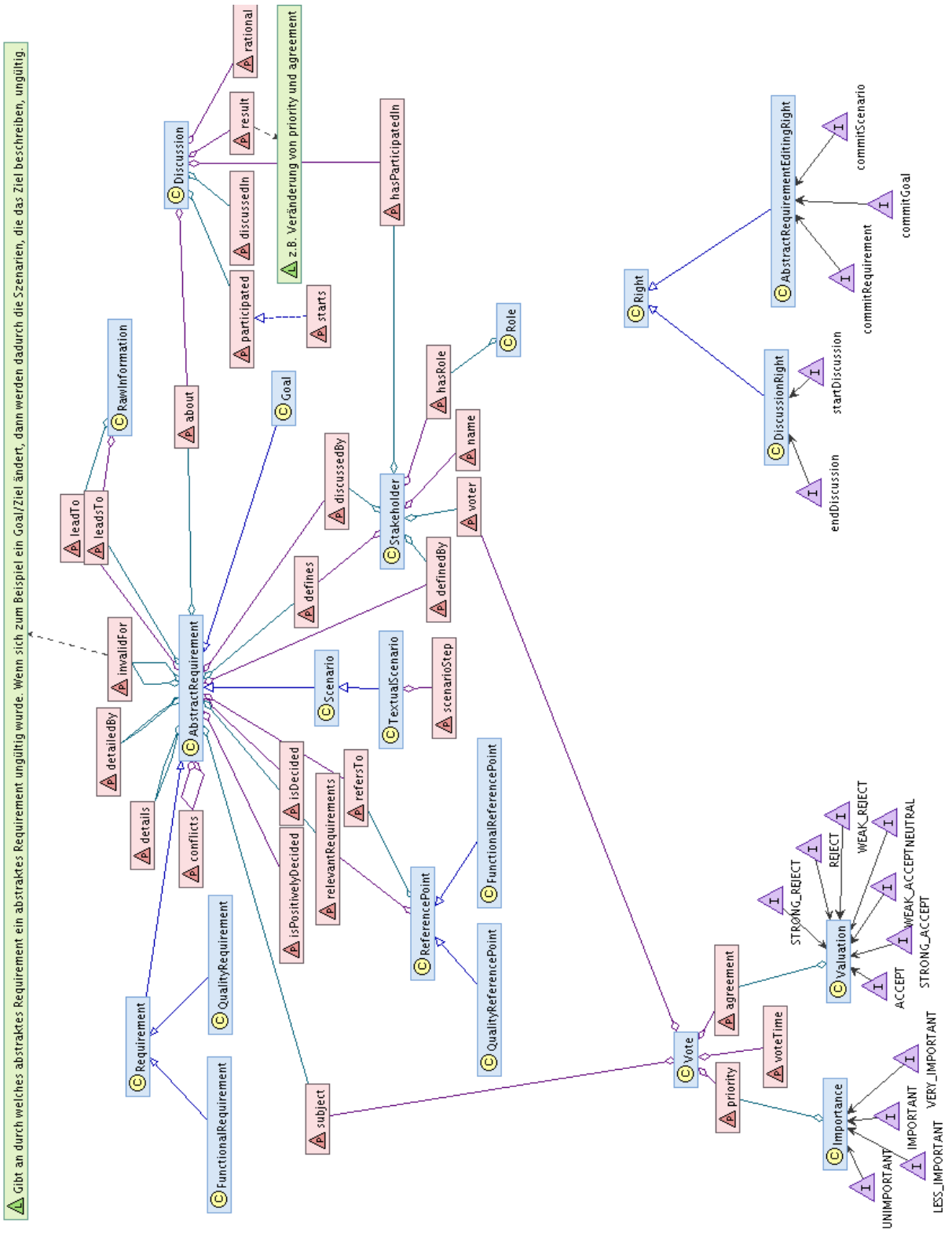
[1] http://jacop.osolpro.com/

of requirements are identified. As mentioned in Chapter 3, requirements engineering is an important part in the system engineering process and therefore this domain is well researched. Riechert et al. [2007] developed the SoftWiki Ontology for RE (SWORE). A part of this ontology is illustrated in Figure 4.1, the whole ontology is online available as RDF-file. The crucial role of the stakeholders, mentioned in Section 3.2.1 is also represented in SWORE. Three key tasks can be identified:

- defining requirements,
- discussing requirements, and
- voting requirements.

Based on these tasks the prototype shall allow the informal input of requirements and the possibility for stakeholders to rate the defined requirements. Since discussing requirements is more relevant for the elicitation and analysis of requirements than for release planning (see Section 3.2) it is not further considered in this master thesis. Having a closer look at the object *AbstractRequirement* of the presented ontology it shows that requirements are not independent from each other. Figure 4.1 illustrates four interdependencies *conflicts, details, detailedBy* and *invalidFor*. However other relations like *dependsOn, entails, isSimilarTo* are offered by SWORE [Lohmann et al., 2008]. For release planning, relations between requirements are crucial, especially those that influence the scheduling. Therefore, a more detailed exploration of this topic is necessary. The literature provides various types and names of requirement interdependencies. The requirement relations implemented in the developed release planner prototype are basically referred to a requirements model proposed by Goknil et al. [2010]. They emphasize four types of requirement relations:

- *Requires relation*: A requirement R_1 requires a requirement R_2 if R_1 is fulfilled only when R_2 is fulfilled. R_2 can be treated as a pre-condition for R_1 [Vicente-Chicote et al., 2007].
- *Refines relation*: A requirement R_1 refines a requirement R_2 if R_1 is derived from R_2 by adding more details to it [van Lamsweerde, 2004].
- *Contains relation*: A requirement R_1 contains requirements $R_2 \dots R_n$ if R_1 is the conjunction of the contained requirements $R_2 \dots R_n$. This relation enables a complex requirement to be decomposed into child requirements [Object Management Group, 2006].
- *Conflicts Relation*: A requirement R_1 conflicts with a requirement R_2 if the fulfillment of R_1 excludes the fulfillment of R_2 and vise versa [Van Lamsweerde et al., 1998].



Source: [2]

Figure 4.1: Part of the SWORE Ontology

[2]http://aksw.org/schema/SWORE/102

Based on this information the users of the prototype shall be able to define the above enumerated relations between the requirements. How these relations can be modelled with CP and their exact behavior are introduced in Section 4.4.
Furthermore, users shall have the possibility to provide additional information. As mentioned in Section 3.3.3 an important part of release planning is the management of the available resources and their assignments to requirements. Each requirement requires a certain amount of different resources. In Section 3.3.3 some examples of typical requirement resources are enumerated. The prototype supports the following ones:

1. developers,
2. testers,
3. budget, and
4. duration in days.

Additionally, to the number of developers and testers, the user shall also be able to assign specific people to a certain requirement. Furthermore, it should be possible to define the basic data of the releases. An important part regarding the definition of releases are the start and end dates of releases. Ruhe [2005] distinguishes two types of release planning:

1. RP with pre-determined time intervals for implementation and
2. planning with flexible intervals.

The prototype shall support both types by allowing not only to define a start and an end date for a certain release, but also to allow defining start and end ranges. Furthermore, for every release the available resources have to be defined.
After a user has defined the basic data for the releases, the human resources, like developers and testers, and definitions of all requirements including their relations and resources, a number of different release plans shall be found. The created plans shall be ranked by their overall stakeholders' satisfaction based on their requirement ratings. If no solution can be found because of insufficient resources or inconsistent requirement relations the reasons shall be diagnosed.
In summary the prototype shall provide the following functionalities:

- Defining human resources (developers, testers).
- Defining a number of releases including their available resources (developers, testers, budget) as well as a start range and an end range.
- Defining requirements including their required resources and their relations to other requirements.
- Calculating a certain number of release plans or all possible release plans.

- Displaying the generated plans sorted by their overall stakeholders' satisfaction.
- Diagnosing inconsistencies in the requirements model.

4.3 Web-Interface For The Release Planner Prototype

4.3.1 Design

This section presents the technologies that are used to develop the release planner prototype. Figure 4.2 illustrates the core components of the system. The user interface is created by the Google Web Toolkit (GWT) and communicates with the server per remote procedure calls (RPC). An Apache Tomcat server handles the RPCs and either stores data to the Sqlite3 database backend or fetches data from it. The actual creation of the release plans is done by the Java Constraints Programming (JaCoP) library[3]. The next four subsections give a short overview of these technologies.

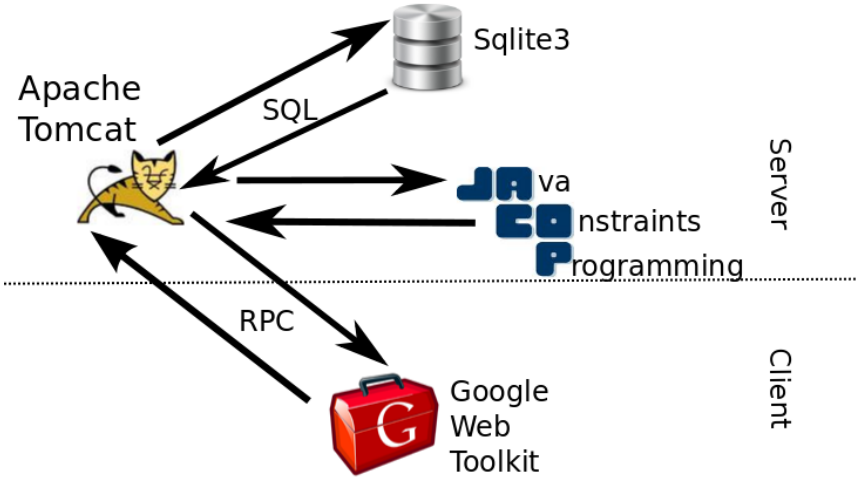


Figure 4.2: Prototype design.

4.3.2 Ajax & Google Web Toolkit

Chapter 3 shows that the requirements engineering process and release planning are influenced by various people. Since they often have different technical background and are often geographically dispersed, the prototype shall be implemented as a web application. Conventional methods that are generating HTML (Hypertext Markup Language) on the server

[3]http://jacop.osolpro.com/

are less flexible and too slow for complex applications. AJAX (Asynchronous JavaScript and XML) is a concept that allows building complex web applications by displacing the representation and some logic from the server to the client. A comparison of the conventional approach and of AJAX is given in Figure 4.3.

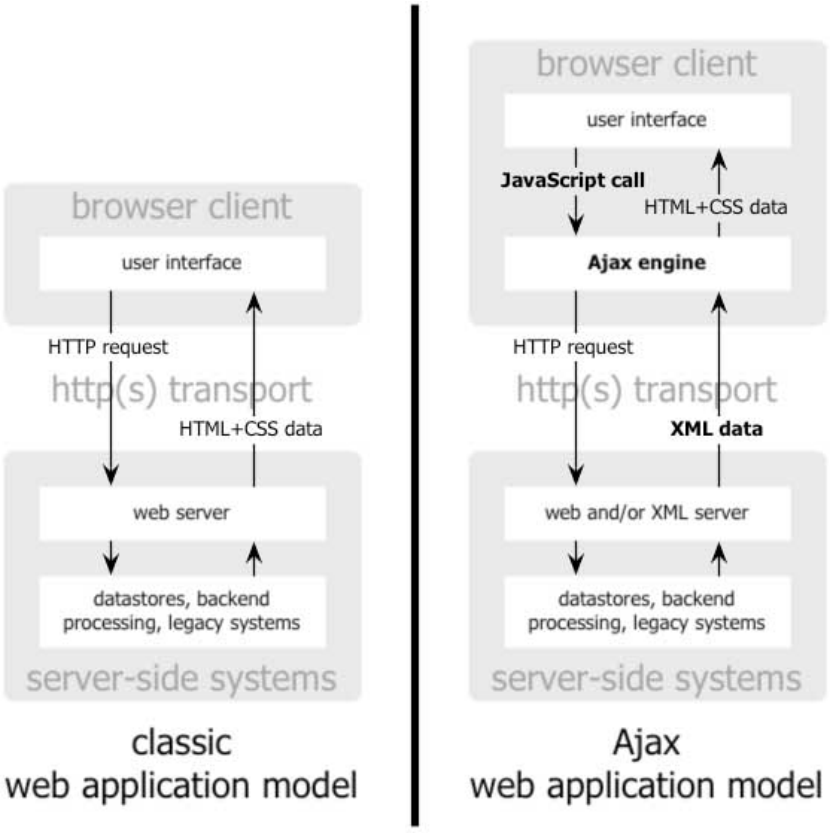


Figure 4.3: Conventional web-applications vs. AJAX [Garrett et al., 2005].

As mentioned before the core technology of AJAX is JavaScript. Unfortunately, JavaScript has some disadvantages like missing integrated development environments (IDE), bad object-oriented support or debugging problems. Especially the last point consumes a lot of time because different browsers and versions produce different errors with more or less useful error messages. Google's Web Toolkit (GWT) claims to solve this problem. This framework basically provides the possibilities of developing the whole client with Java instead of writing native JavaScript code. In addition to the most common Java libraries a GUI library similar to SWING is available. In recent years GWT became more and more popular and some external frameworks including frequently used widgets and components were developed. One of the most powerful is GWT-ext[4] and therefore it is

[4]http://code.google.com/p/gwt-ext/

used to develop the prototype. During the developing process this framework has been integrated into another one, called SmartGWT[5]. In addition to existing libraries, the use of popular IDEs is now also possible, including all the advantages like syntax highlighting and code completion. Since browsers are not able to execute Java code, it is translated to JavaScript by GWT to deploy a project. This allows generating optimized code for different browsers. It is also possible to set breakpoints in the original Java code and debug it in a common browser like Firefox. Since a plugin for the Eclipse[6] IDE is available that supports the aforementioned features, it is used to develop the prototype. The communication between server and clients can be done by custom HTTP requests or the provided RPCs. The first method is especially useful when the server is not using Java. Since the prototype is using an Apache Tomcat server (see next Section 4.3.3) the communication is done by the provided RPC framework. Remote procedure calls are a well researched topic in informatics and were introduced by White [1976]. A quite simple explanation of this technique is given by Birrell and Nelson [1984]. They mention that RPCs are inspired by the idea of common procedure calls on a single computer, but transferring the data across a communication network. When a RPC is invoked, the calling process is suspended and the called procedure is executed on the remote environment. After completion, the results are sent back to the caller and the procedure can be continued. During this call other procedures are able to run.
GWT RPCs also work this way and allow the user to continue working during a server communication process. Defining such a procedure call needs just an interface definition on the client side and an implementation of this interface on the server. Afterwards the client can call server methods similar to local ones with the exception that a so-called asynchronous callback function has to be passed as parameter. This is necessary to tell the client what to do after the server has responded. A more detailed description about communication and other features of GWT can be found on the associated website[7].

4.3.3 Apache Tomcat

As described in the section above, the web-based client of the prototype uses RPCs for the communication with a server. Therefore, the open source servlet container Apache Tomcat[8] is used. Servlets can be seen as the Java pendant to conventional common gateway interfaces (CGI). A servlet is a Java class that implements the Java Servlet API, a protocol to respond to HTTP requests. It allows to remember states and session variables across

[5]http://code.google.com/p/smartgwt/
[6]http://www.eclipse.org/
[7]http://code.google.com/webtoolkit/overview.html
[8]http://tomcat.apache.org/

multiple server transactions. Compared to simple Java CGI scripts it improves performance by executing every request as an own thread instead of creating a new process. Another advantage of using the Apache Tomcat server is the ease of deploying GWT projects.

4.3.4 Sqlite3

Sqlite3 is chosen to store the user input and the configuration of the release planner. It is a relational database system and implemented in C. Unlike the most database systems Sqlite3 does not need to run on a server. Hence, it is frequently used in embedded systems. It is easy to use on different platforms and provides most of the SQL-92 standard[9]. Even though Sqlite3 does neither support synchronous write operations nor allows user permissions management, it provides the most important features for small applications or in our case a prototype. Sqlite3 provides a large number of interfaces for different programming languages. It also provides an implementation of the Java database connectivity (JDBC). This is an interface for Java that standardizes database connections, queries, and results of relational databases.

4.3.5 JaCoP Library

In Section 2 an introduction to constraint programming is given and some libraries are presented that allow modelling constraint satisfaction problems. Since the prototype is developed in Java, the JaCoP library is used to model a release planning task. The development of JaCoP started 2001 by Krzysztof Kuchcinski and Radoslaw Szymanek and is still in progress. The implementation of JaCoP was influenced by a lot of research work in the area of CP. Their developers are still trying to keep the library up to date by extending and improving it, based on current researches. In the last years also a lot of research studies have used JaCoP. A detailed list of the underlying research works and studies using the library can be found on the project's website[10]. The release 2.4 used for this master thesis provides all constraints described in Section 2.4.6 as well as a number of global constraints including the ones of Section 2.4.7. The usage of the library and how release plans can be generated with JaCoP is shown in Section 4.4.

[9] http://www.contrib.andrew.cmu.edu/ shadow/sql/sql1992.txt
[10] http://www.osolpro.com/jacoptwiki/bin/view.pl/Main/WebHome

4.4 Release Planning With JaCoP

This section gives a detailed description of how release plans are generated by the JaCoP library. It presents how a solution looks like, describes the used variables and constraints, and finally how a solution can be found.
For a better understanding of the used variables and constraints a small requirements model is introduced. Based on this example all variables and constraints are described. Several code snippets are presented to show how constraints are generated in the release planner. Additionally the generated constraints of the proposed example are displayed in tables. Table 4.1 and Table 4.2 show the basic data of the example requirements model. In Section 4.4.3 some relations between the requirements of Table 4.1 will be added.

Id	Name	#Devel.	#Testers	Budget	Duration	Latest Release
Req1	Authentication	Abstract				1
Req2	Apache Authentication	1	2	1000	3	—
Req3	Database Authentication	Abstract				—
Req4	Auth. Tables	2	1	0	2	—
Req5	Auth. DB Communication	1	1	0	3	—
Req6	Auth. Password Encryption	2	1	0	3	—
Req7	Database Server	1	2	300	3	—

Table 4.1: Requirements to explain the used variables and constraints.

Id	Name	Start range	End range	#Dev.	#Test.	Budget	Max. Days
Rel1	Release 1	01.03.2010 - 01.03.2010	05.03.2010 - 08.03.2010	3	2	500	6
Rel2	Release 2	08.03.2010 - 09.03.2010	11.03.2010 - 11.03.2010	2	1	0	4

Table 4.2: Releases to explain the used variables and constraints.

4.4.1 Solutions

Before describing variables, constraints, and searching for plans, the expected result should be determined. Generally there are two types of information that are necessary for a useful release plan. The first one are the start and end dates of the releases. In Section 3.3 it was mentioned that the prototype shall not only support requirements scheduling to properly defined release dates but also allow flexible release starts and ends. Hence,

for every release a start and an end has to be determined. The second part of a result are the assignments of the requirements to the releases. In the next sections it is shown that scheduling a requirement to a particular release is not enough. Some constraints like resources or relations between requirements need a much more detailed scheduling. The prototype solves this problem by not only assigning a start and an end date to the releases but also by determining exact start and end dates for all requirement implementations. In summary a result consists of the start and end times of each release and of the start and end times of all requirement implementations. In Chapter 5 a solution is presented as an appointment calendar. Unfortunately this format is inappropriate to model the scheduling problem. Therefore all dates are converted to days, ranging from 0 to n. Whereby, 0 means the first day of a project and n the last possible day. This has to be done for all date inputs. The generated solutions are finally converted back to dates for presenting the plans in a time table. Assuming that r is a release, n the number of releases, q a requirement, and m the number of requirements, a solution is defined in Formula 1. A

$$r_{\text{start}_1} r_{\text{end}_1} r_{\text{start}_2} r_{\text{end}_2} \dots r_{\text{start}_n} r_{\text{end}_n} q_{\text{start}_1} q_{\text{end}_1} q_{\text{start}_2} q_{\text{end}_2} \dots q_{\text{start}_m} q_{\text{end}_m}$$

Formula 1: Solution for a release planning task.

Rel1	Rel2	Req1	Req2	Req3	Req4	Req5	Req6	Req7
0 6	6 9	6 6	9 12	6 6	3 5	3 6	6 9	0 3

Table 4.3: Possible solution of the example requirements model.

valid release plan for the presented requirements model is shown in Table 4.3 and illustrated in Figure 4.4. The first four numbers of Table 4.3 represent the start and end day of the releases, while the last 14 numbers indicate the start and end dates of the requirement implementations. This quite simple sequence of numbers is enough to represent a complete release plan. All other used variables and constraints influence the start and end times but not the representation of a solution. The generated plan shows two characteristics. First the requirements “Authentication” (Req1) and “Database Authentication” (Req3) have equal start and end times. Since these requirements are marked as abstract, see Table 4.1, they do not need any resources. In Section 4.4.3 it is shown that the scheduling of such requirements depends on their relations and not on their resources. Another specific characteristic is the scheduling of “Apache Authentication” (Req2) beyond the second release. It should be pointed out that some requirements do not have to be implemented. This can have different reasons like incompatibilities to others or they may be optional, see Section 4.4.3. Since a solver searches for a start and an end date of every requirement implementation, rejected ones are simply scheduled after the end of the last release.

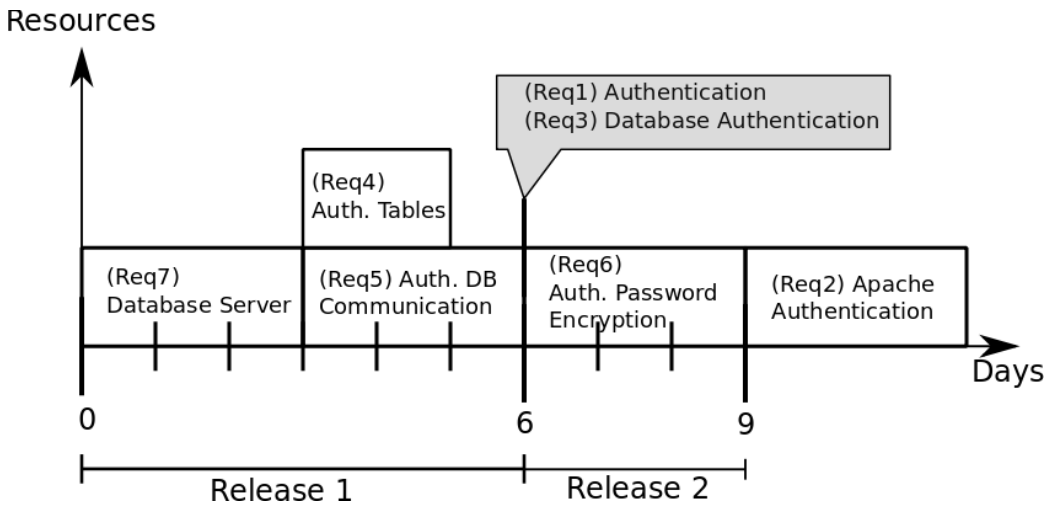


Figure 4.4: Graphical representation of a release plan.

4.4.2 Variables

CP variables in JaCoP can be defined by a finite domain variable (FDV). In the explanations below the term variable is used as a Java variable, whereas FDV means a variable in the context of CP. The library offers different constructors to create an instance of a FDV. The most frequently one is:

FDV(Store store, java.lang.String name, int min, int max);.

- *Store store* the created FDV instance will be stored to this store.
- *String name* a human readable name of the variable.
- *int min* the minimum value in the domain of the variable.
- *int max* the maximum value in the domain of the variable.

In most cases not only one FDV is generated, but for every requirement or for every release a particular FDV is created. Therefore, the most FDVs are stored in array variables. The explanations follow the syntax that arrays are named in plural and single variables in singular. The most important FDVs are listed in Tables 4.4, 4.7, and 4.8. A solution includes start and end times of every release and of every requirement implementation. Table 4.4 presents these variables and the associating FDVs. Table 4.5 presents the FDVs of the start and end times of every release. The domains of these FDVs are determined by the user-defined release ranges. Since in the example, see Table 4.1, the first release is identical to the project start and it can only start on the 1st of March, the domain is 0. The release takes 5 or 6 days and therefore, it has to end on the 5th or on the 6th day. It shall be mentioned that only workdays are considered. Therefore, the 6th and

Name	Description	FDV description
release_starts	One FDV per release	One FDV defines the day a release starts. The FDV's domain depends on an user-defined start range.
release_ends	One FDV per release	One FDV defines the day a release ends. As for the FDVs of variable release_starts the domain depends on an user-defined range.
impl_starts	One FDV per requirement	One FDV defines the day a requirement implementation starts. The domain ranges from 0 (project start) to the latest possible start day.
impl_ends	One FDV per requirement	One FDV defines the day a requirement implementation ends. The domain ranges from the duration of the requirement to the latest possible end day.

Table 4.4: Solution variables.

Id	Name	Start	Duration	End
Rel1	Release 1	0	{5..6}	{5..6}
Rel2	Release 2	{5..6}	{3..4}	9

Table 4.5: Domains of the release FDVs.

Id	Name	Start	Duration	End
Req1	Authentication	{0..24}	0	{0..24})
Req2	Apache Authentication	{0..21}	3	{0..24})
Req3	Database Authentication	{0..24}	0	{0..24})
Req4	Auth. Tables	{0..22}	2	{0..24})
Req5	Auth. DB Communication	{0..21}	3	{0..24})
Req6	Auth. Password Encryption	{0..21}	3	{0..24})
Req7	Database Server	{0..21}	3	{0..24})

Table 4.6: Domains of the requirement FDVs.

Name	Description	FDV description
bounded_developers	One FDV per requirement	One FDV represents to which extent developers are already assigned to other requirements when this one is scheduled. This means that these developers are not available for the associated requirement. The domain of the FDV can range from 0 to the maximum number of developers.
needed_developers	One FDV per requirement	This variable's domain is set to a single value, the needed developers for the associated requirement. Actually it can be seen as constant but some global constraints require FDVs as parameter.
used_developers	One FDV per requirement	The value of this FDV is the result of adding the values of the corresponding FDVs *bounded_developers* and *needed_developers*.

Table 4.7: Resource variables.

7^{th} of March are ignored. The second release can start on the 6^{th} or on the 7^{th} day, it can take 3 or 4 days, and it has to end on the 9^{th} day. For a better understanding why the start times 0, 5, and 6 represent the first, the 6^{th} and the 7^{th} day see Section 4.4.1, especially Figure 4.4. In Table 4.6 the domains of the start times, durations, and end times of the seven requirement implementations are shown. The end time for each requirement implementation is calculated by adding the durations of all requirements $(0+3+0+2+3+3+3=14)$ plus the maximum days the releases can take $(6+4=10)$. Therefore, the end time FDVs have all the same domain, ranging from 0 to 24. The domains of the start times are equal to the domains of the end times, but reduced by their own duration.

Beside the variables that represent the actual solution, other types of FDVs are necessary to model a release planner. A goal of the prototype is to schedule the requirements considering the resources developers, testers, and budget. The FDVs for all resources can be defined in the same way. Important for modelling the problem is that every resource has a *used*, *needed*, and a *bounded* FDV. Therefore, the following variables are used:

- used_developers, used_testers, used_budget
- needed_developers, needed_testers, needed_budget
- bounded_developers, bounded_testers, bounded_budget

Table 4.7 is describing these three types with the help of the developer resource. In the next section it can be seen that some constraints require auxiliary FDVs. Therefore a few more variables are introduced in Table 4.8.

Name	Description	FDV description
durations	One FDV per requirement	Represents the duration of a requirement implementation. As the *needed* FDVs of Table 4.7 this FDV is constant but needs to be declared as FDV, because of some constraints.
release_durations	One FDV per release	One FDV defines the duration of a release. The domain ranges from the earliest day a release can start to the latest day it can end.
project_end	Unique	Is equal to the end date of the last release.
assigned_releases	One FDV per requirement	Represents the release to which a requirement is assigned. The domain ranges from 0 to the number of releases. 0 means that the requirement is assigned to the first release and the maximum value means that the requirement shall not be implemented.
release_durations_of _requirements	One FDV per requirement	One FDV represents the release's duration to which the requirement is assigned.
release_start_of _requirements	One FDV per requirement	One FDV represents the release's start day to which the requirement is assigned.

Table 4.8: Auxiliary variables.

For a better understanding of the constraints presented in the next section, all generated constraints of the presented example are shown. Since every constraint affects at least one FDV, a notation is required to represent the access to a particular FDV. The used FDVs are displayed by the IDs introduced in Table 4.5 and Table 4.6, followed by "→" and the name of the used FDV. For example, accessing the start time FDV of requirement "Apache Authentication" (Req2) looks like this:

Req2→start

4.4.3 Constraints

This subsection introduces the core of every CSP, the constraints. The JaCoP library offers a wide range of constraints, including primitive, logical, and conditional constraints, as well as global ones. Each constraint is a class derived from the super class *Constraint*. The constraints of this work follow three different purposes. The first one deals with defining the dependencies between the variables introduced in Section 4.4.2 and it presents some constraints that arrange basic conditions of a release plan. The second part of this section introduces constraints that are used to handle the required and available resources. Finally, the relations mentioned in Section 4.2 are described in detail.

Authentication:	XplusYeqZ(Req1→start, Req1→duration, Req1→end)
Apache Authentication:	XplusYeqZ(Req2→start, Req2→duration, Req2→end)
DB Auth.:	XplusYeqZ(Req3→start, Req3→duration, Req3→end)
Auth. Tables:	XplusYeqZ(Req4→start, Req4→duration, Req4→end)
Auth. DB Communication:	XplusYeqZ(Req5→start, Req5→duration, Req5→end)
Auth. Pwd Encryption:	XplusYeqZ(Req6→start, Req6→duration, Req6→end)
Database Server:	XplusYeqZ(Req7→start, Req7→duration, Req7→end)
Release 1:	XplusYeqZ(Rel1→start,Rel1→duration,Rel1→end)
Release 2:	XplusYeqZ(Rel2→start,Rel2→duration,Rel2→end)

Table 4.9: Start, duration, and end constraints for the presented example.

Basic Conditions

First of all some constraints have to be imposed to express the relation between start day, duration, and end day. This has to be done for all requirements and all releases. Therefore, the first primitive constraint of the JaCoP library can be introduced. For this purpose the constraint *XplusYeqZ* is used. All primitive constraints in JaCoP reveal the type of arguments and provide information about their usage by their names. The capitals X, Y, and Z indicate that three FDVs are expected as arguments. A capital C in the name means that a constant, more precisely an integer is expected. Therefore, the now used constraint takes three FDVs as parameters and ensures that the value of the first argument plus the value of the second argument is equal to the third one. The relations between start day, duration, and end day are created by the code presented in Listing 4.1. Table 4.9 shows the constraints generated in presented example.

```
1 for(int req = 0; req < num_requirements; req++)
2  addConstraint(new XplusYeqZ(impl_starts[req],durations[req],impl_ends[req]));
3 for(int rel = 0; req < num_releases; rel++)
4  addConstraint(new XplusYeqZ(release_starts[rel],release_durations[rel],release_ends[rel]));
```

Listing 4.1: Constraints for start, duration, and end of requirement implementations and releases.

Listing 4.2 introduces two global constraints, *Max* and *Min*. Both constraints takes a FDV as first argument and an array of FDVs as second one. *Max* ensures that the first argument's value is equal to the maximum value of the second argument. *Min* is the counterpart of *Max*, it assigns the minimum value of the second argument to the first one. In line 1 of Listing 4.2 *Max* is used to define the end of a project. It shows that the end of

a project is equal to the end of the last release. Later in this section this FDV is often used by constraints that handle requirements that shall not be implemented. Table 4.10 shows how this constraint looks in the presented example. It is predictable that the value 9 will be assigned to the FDV project_end, because of the domains of both release end FDVs. Having a closer look at Table 4.5 reveals that the end time FDV of the first release can not exceed the end time FDV of the second release. Since the second release can only end on day 9, the FDV project_end has also to be 9.

Min is used in lines 2 and 3 of Listing 4.2 to guarantee that a release and at least one requirement implementation have to start on the first day of a project. Furthermore, a release shall not start before the previous one has ended. Therefore, the primitive constraint *XgteqY* is used, in Line 5 of Listing 4.2. The name of this constraint already indicates that two FDVs are expected and it ensures that the first one is greater or equal to the second one. Since the presented example consists only of two releases, only one constraint is necessary, see Table 4.10.

```
1 addConstraint(new Max(project_end, release_ends))
2 addConstraint(new Min(new Variable(store,0,0), impl_starts))
3 addConstraint(new Min(new Variable(store,0,0), release_starts))
4 for(int rel = 1; rel < num_releases; rel++)
5   addConstraint(new XgteqY(release_starts[rel],release_ends[rel-1]))
```

Listing 4.2: Constraints for the start and the end of a project.

Project End:	Max(project_end,{Rel1→end,Rel2→end})
At least one requirement starts on day 0 :	Min(0, {Req1→start, Req2→start, Req3→start, Req4→start, Req5→start, Req6→start, Req7→start})
At least one release starts on day 0 :	Min(0, {Rel1→start, Rel2→start})
Release 2 starts after release 1:	XgteqY(Rel2→start,Rel1→end)

Table 4.10: Determining the project end, ensuring that one release and at least one requirement implementation starts on the first day, and releases must not overlap.

Another aspect that has to be considered is the determination of the start and end days of the releases. As mentioned in 4.3.1 these dates have to be within an user-defined range. The constraints, generated in Listing 4.3, reduce these domains. A release starts when the first requirement implementation of this release starts. It does not make sense to start a release on day 25, but the first requirement implementation of this release is scheduled to day 27. Similar to the release start, a release shall end immediately after its last requirement was implemented. Therefore, a release start has to be equal to a requirement start and a release

end has to be equal to a requirement end. However, if all requirement implementations of a release are completed before the user-defined end range, the release end shall be set to the first day of this range. For this purpose the *Or* constraint can be used. It takes a list of primitive constraints and ensures that at least one of them is true. Listing 4.3 presents the implementation of these conditions. In lines 7 and 8 the possible release starts and ends are defined and stored in arrays. This is done by creating the primitive constraint *XeqY*, that ensures that the first argument is equal to the second one. In line 8 the first primitive constraint, using a constant as argument is introduced. *XeqC* works equal to *XeqY* but excepts an integer, instead of an FDV as second argument. In line 10 this constraint is used to define that a release can also end on the first day of the user-defined end range. In the presented example this would be day 5 for the first release and day 9 for the second one. In lines 11 and 12 the logical disjunctions between the created constraints are imposed and ensure that at lease one of the created constraints is true. Table 4.11 shows these constraints for the presented example.

```
1  for(int rel = 0; rel < num_releases; rel++)
2  {
3    ArrayList<PrimitiveConstraint> possible_release_ends = new ArrayList<
         PrimitiveConstraint>();
4    ArrayList<PrimitiveConstraint> possible_release_starts = new ArrayList<
         PrimitiveConstraint>();
5    for(int req = 0; req <num_requirements; req++)
6    {
7      possible_release_ends.add(new XeqY(release_ends[rel], impl_ends[req]));
8      possible_release_starts.add(new XeqY(release_starts[rel], impl_starts[req]));
9    }
10   possible_release_ends.add(new XeqC(release_ends[rel],release_end_range_begin[rel]));
11   addConstraint(new Or(possible_release_ends))
12   addConstraint(new Or(possible_release_starts))
13 }
```

Listing 4.3: Constraints for release starts and ends.

Since there are some constraints that limit release starts and ends, it is obvious that such constraints also exist for requirement implementations. It has to be ensured that a requirement implementation shall start as soon as enough resources are available and all conditions are fulfilled. Therefore, a requirement implementation can only start at the beginning of a release or after another requirement has been completed. One exception is valid for requirements that shall not be implemented. These shall be scheduled after the end of the project. The code, shown in Listing 4.4, creates these constraints. The constraints created in lines 4 to 8 express that an implementation can start after another one is completed. Of course, a requirement can also be implemented at the beginning of a release, this is taken into account in line 10. Line 11 ensures that rejected requirement

Possible starts of release 1:	Or(XeqY(Rel1→start,Req1→start), XeqY(Rel1→start,Req2→start),XeqY(Rel1→start,Req3→start), XeqY(Rel1→start,Req4→start),XeqY(Rel1→start,Req5→start), XeqY(Rel1→start,Req6→start),XeqY(Rel1→start,Req7→start),	Line 8
Possible starts of release 2:	Or(XeqY(Rel2→start,Req1→start), XeqY(Rel2→start,Req2→start),XeqY(Rel2→start,Req3→start), XeqY(Rel2→start,Req4→start),XeqY(Rel2→start,Req5→start), XeqY(Rel2→start,Req6→start),XeqY(Rel2→start,Req7→start),	Line 8
Possible ends of release 1:	Or(XeqY(Rel1→end,Req1→end), XeqY(Rel1→end,Req2→end),XeqY(Rel1→end,Req3→end), XeqY(Rel1→end,Req4→end),XeqY(Rel1→end,Req5→end), XeqY(Rel1→end,Req6→end),XeqY(Rel1→end,Req7→end), XeqC(Rel1→end,5))	Line 7 Line 10
Possible ends of release 2:	Or(XeqY(Rel2→end,Req1→end), XeqY(Rel2→end,Req2→end),XeqY(Rel2→end,Req3→end), XeqY(Rel2→end,Req4→end),XeqY(Rel2→end,Req5→end), XeqY(Rel2→end,Req6→end),XeqY(Rel2→end,Req7→end), XeqC(Rel2→end,9))	Line 7 Line 10

Table 4.11: Start and end constraints of the releases of the presented example.

implementations can be scheduled after the end of the project. To sum up, requirements can start at the beginning of a release, after another requirement implementation ends or after the end of the project. Referring to the presented example every requirement has 9 possibilities to start: 6 requirement implementation ends, 2 release starts or after the end of the project. The generated constraints, imposed in line 12 of Listing 4.4, are shown in Table 4.12.

```
1  for(int req = 0; req < num_requirements; i++}
2  {
3    ArrayList<PrimitiveConstraint> requ_starts = new ArrayList<PrimitiveConstraint>();
4    for(int tmp_req=0; tmp_req<num_requirements;tmp_req++)
5    {
6      if(req!=tmp_req)
7        requ_starts.add(new XeqY(impl_starts[req],impl_ends[tmp_req]));
8    }
9    for(int rel=0;rel<num_releases;rel++)
10     requ_starts.add(new XeqY(impl_starts[req], release_starts[rel]));
11 requ_starts.add(new XeqY(impl_starts[req],project_end));
12 addConstraint(new Or(requ_starts));
13 }
```

Listing 4.4: Constraints for requirement implementation starts and durations.

Since requirements that shall not be implemented are scheduled to the end of the project, it has to be ensured that this does not happen to mandatory ones. In Listing 4.5 it can be seen that it depends on the relations of a requirement, if it is mandatory or not. Later in this Section it is described why requirements that are connected to others by *incompatible*, *is_a* or *part_of* have not to be implemented in any case. Therefore, the constraint in line 7 is only generated for requirements that do not have such relations. Since there

Possible starts of "Authentication":	Or(	
	XeqY(Req1→start,Req2→end),XeqY(Req1→start,Req3→end),	Line 4-8
	XeqY(Req1→start,Req4→end),XeqY(Req1→start,Req5→end),	
	XeqY(Req1→start,Req6→end),XeqY(Req1→start,Req7→end),	
	XeqY(Req1→start,Rel1→start),XeqY(Req1→start,Rel2→start),	Line 9-10
	XeqY(Req1→start,project_end))	Line 11
Possible starts of "Apache Authentication":	Or(	
	XeqY(Req2→start,Req1→end),XeqY(Req2→start,Req3→end),	Line 4-8
	XeqY(Req2→start,Req4→end),XeqY(Req2→start,Req5→end),	
	XeqY(Req2→start,Req6→end),XeqY(Req2→start,Req7→end),	
	XeqY(Req2→start,Rel1→start),XeqY(Req2→start,Rel2→start),	Line 9-10
	XeqY(Req2→start,project_end))	Line 11
Possible starts of "Database Authentication":	Or(	
	XeqY(Req3→start,Req1→end),XeqY(Req3→start,Req2→end),	Line 4-8
	XeqY(Req3→start,Req4→end),XeqY(Req3→start,Req5→end),	
	XeqY(Req3→start,Req6→end),XeqY(Req3→start,Req7→end),	
	XeqY(Req3→start,Rel1→start),XeqY(Req3→start,Rel2→start),	Line 9-10
	XeqY(Req3→start,project_end))	Line 11
Possible starts of "Auth. Tables":	Or(	
	XeqY(Req4→start,Req1→end),XeqY(Req4→start,Req2→end),	Line 4-8
	XeqY(Req4→start,Req3→end),XeqY(Req4→start,Req5→end),	
	XeqY(Req4→start,Req6→end),XeqY(Req4→start,Req7→end),	
	XeqY(Req4→start,Rel1→start),XeqY(Req4→start,Rel2→start),	Line 9-10
	XeqY(Req4→start,project_end))	Line 11
Possible starts of "Auth. DB Communication":	Or(	
	XeqY(Req5→start,Req1→end),XeqY(Req5→start,Req2→end),	Line 4-8
	XeqY(Req5→start,Req3→end),XeqY(Req5→start,Req4→end),	
	XeqY(Req5→start,Req6→end),XeqY(Req5→start,Req7→end),	
	XeqY(Req5→start,Rel1→start),XeqY(Req5→start,Rel2→start),	Line 9-10
	XeqY(Req5→start,project_end))	Line 11
Possible starts of "Authent. Password Encryption":	Or(	
	XeqY(Req6→start,Req1→end),XeqY(Req6→start,Req2→end),	Line 4-8
	XeqY(Req6→start,Req3→end),XeqY(Req6→start,Req4→end),	
	XeqY(Req6→start,Req5→end),XeqY(Req6→start,Req7→end),	
	XeqY(Req6→start,Rel1→start),XeqY(Req6→start,Rel2→start),	Line 9-10
	XeqY(Req6→start,project_end))	Line 11
Possible starts of "Database Server":	Or(	
	XeqY(Req7→start,Req1→end),XeqY(Req7→start,Req2→end),	Line 4-8
	XeqY(Req7→start,Req3→end),XeqY(Req7→start,Req4→end),	
	XeqY(Req7→start,Req5→end),XeqY(Req7→start,Req6→end),	
	XeqY(Req7→start,Rel1→start),XeqY(Req7→start,Rel2→start),	Line 9-10
	XeqY(Req7→start,project_end))	Line 11

Table 4.12: Constraints for requirement implementation starts.

are no relations defined for the example yet, this constraint would be generated for all seven requirements of Table 4.1. In anticipation of the relations introduced in Section 4.4.3 and illustrated in Figure 4.6 the constraints shown in Table 4.13 are created. Since "Authentication" and "Database Server" are the only requirements that have no outgoing

is_a, *part_of*, and *incompatible* relations, they are mandatory.

```
1  for(int req = 0; req < num_requirements; req++)
2  {
3    Requirement requirement = getRequirement(req);
4    if(!requirement.hasRelation(RelationConstant.INCOMPATIBLE) &&
5      !requirement.hasRelation(RelationConstant.IS_A) &&
6      !requirement.hasRelation(RelationConstant.PART_OF))
7        addConstraint(new XltC(assigned_release_[req],num_releases_));
8  }
```

Listing 4.5: Mandatory requirements have to be implemented.

'Authentication' is mandatory	XltC(Req0→assigned_release,2)
'Database Server' is mandatory	XltC(Req6→assigned_release,2)

Table 4.13: Mandatory requirements.

The last Listing 4.6 of this section shows an additional feature of the prototype. This constraint ensures that a requirement has to be implemented in a particular release or earlier. In Table 4.1 it can be seen that "Authentication" shall be implemented in the first release. Therefore only one constraint is imposed in the presented example, see Table 4.14. Since the domain of the assigned_releases FDVs starts from 0 and not from 1, the latest possible release is decremented by one.

```
1  private void addLatestReleaseConstraint(req, latest_release)
2  {
3    addConstraint(new XlteqC(assigned_releases[req], (latest_release-1)));
4  }
```

Listing 4.6: Ensures that a requirement is implemented at a particular release.

"Authentication" is finished at the first release:	XlteqC(Req1→assigned_release, 0);

Table 4.14: Req1 has to be scheduled to the first release.

Resource Constraints

The second type of constraints refers to scheduling requirements in a way that a plan does not exceed the available resources at any time. First the relations between already bounded, needed, and finally used resources have to be defined. It was already mentioned before, that the bounded resources plus needed ones results in used resources. Listing 4.7 shows the code that creates these relations. It can be seen that this has to be done for all resources and for all requirements. Therefore, 21 constraints are created in the presented example, see Table 4.15. Later in this section it can be seen that these constraints are necessary to limit the resources of a release.

```
1 for(int req = 0; req < num_requirements; req++)
2 {
3   addConstraint(new XplusYeqZ(bounded_developers[req],needed_developer[req],used_developers[req]));
4   addConstraint(new XplusYeqZ(bounded_testers[req],needed_tester[req],used_testers[req]));
5   addConstraint(new XplusYeqZ(bounded_budget[req],needed_budget[req],used_budget[req]));
6 }
```

Listing 4.7: Constraints for bounded, needed, and used resources.

Authentication:	XplusYeqZ(Req1→bounded_developers,Req1→needed_developers,Req1→used_developers) XplusYeqZ(Req1→bounded_testers,Req1→needed_testers,Req1→used_testers) XplusYeqZ(Req1→bounded_budget,Req1→needed_budget,Req1→used_budget)
Apache Authentication:	XplusYeqZ(Req2→bounded_developers,Req2→needed_developers,Req2→used_developers) XplusYeqZ(Req2→bounded_testers,Req2→needed_testers,Req2→used_testers) XplusYeqZ(Req2→bounded_budget,Req2→needed_budget,Req2→used_budget)
Database Authentication:	XplusYeqZ(Req3→bounded_developers,Req3→needed_developers,Req3→used_developers) XplusYeqZ(Req3→bounded_testers,Req3→needed_testers,Req3→used_testers) XplusYeqZ(Req3→bounded_budget,Req3→needed_budget,Req3→used_budget)
Auth. Tables:	XplusYeqZ(Req4→bounded_developers,Req4→needed_developers,Req4→used_developers) XplusYeqZ(Req4→bounded_testers,Req4→needed_testers,Req4→used_testers) XplusYeqZ(Req4→bounded_budget,Req4→needed_budget,Req4→used_budget)
Auth. DB Communication	XplusYeqZ(Req5→bounded_developers,Req5→needed_developers,Req5→used_developers) XplusYeqZ(Req5→bounded_testers,Req5→needed_testers,Req5→used_testers) XplusYeqZ(Req5→bounded_budget,Req5→needed_budget,Req5→used_budget)
Auth. Password Encryption:	XplusYeqZ(Req6→bounded_developers,Req6→needed_developers,Req6→used_developers) XplusYeqZ(Req6→bounded_testers,Req6→needed_testers,Req6→used_testers) XplusYeqZ(Req6→bounded_budget,Req6→needed_budget,Req6→used_budget)
Database Server:	XplusYeqZ(Req7→bounded_developers,Req7→needed_developers,Req7→used_developers) XplusYeqZ(Req7→bounded_testers,Req7→needed_testers,Req7→used_testers) XplusYeqZ(Req7→bounded_budget,Req7→needed_budget,Req7→used_budget)

Table 4.15: Constraints to define the relations between bounded, needed, and used resources.

The challenge of this task is to ensure that the available resources of a release are never exceeded by the assigned requirement implementations. In Section 2.3.3 a simple scheduling problem was solved by the *Cumulative* constraint. It seems obvious that this constraint can also help to solve this problem. Listing 4.8 shows how this constraint could be used for the resources developers and testers.

```
1  Cumulative(impl_starts, durations, needed_developers, new Variable(store,
       max_developer_int, max_developer_int)))
2  Cumulative(impl_starts, durations, needed_testers, new Variable(store, max_tester_int,
       max_tester_int)))
```

Listing 4.8: Cumulative constraints for developers and testers.

These constraints ensure that requirements that shall be implemented at the same time do not exceed the maximum number of available developers and testers. The usage of this constraint was already described in detail in Section 2.3.3. In the case of the release planner the first three arguments are arrays that store the start times, the durations, and the required resources of all requirements. The last argument defines the threshold that must not be exceeded. The code snippet above shows that this threshold has to be passed as FDV. Unfortunately, this threshold is valid for the whole project, but in release plans the limits differ from release to release. In Figure 4.5 the cumulative constraints of the developer resource is displayed by a red dotted line. There the upper threshold of available developers is set to 3, which is the number of available developers of the first release, see Table 4.2. As expected this would work for the first release, but it can also be seen that this constraint is not sufficient for the second release. The constraints above do not consider that the available resources depend on the release to which a requirement is assigned. Therefore, it is necessary to model the resources in a different way. This can be done with the global constraint *diff2* and the bounded, needed, and used variables, see Table 4.7.

```
1  Diff2(store, impl_starts, bounded_developers, durations, needed_developer)
2  Diff2(store, impl_starts, bounded_testers, durations, needed_tester)
3  Diff2(store, release_starts_of_requirements, bounded_budget,
       release_durations_of_requirements, needed_budget)
```

Listing 4.9: Diff2 constraints for developers, testers, and budget.

Before describing how this constraint is used in the planner model, a short explanation of it shall be given. The guide from the JaCoP website describes it as follows:

> "Diff2 constraint takes as an argument a list of 2-dimensional rectangles and assures that for each pair i,j ($i \neq j$) of such rectangles, there exists at least one dimension k where i is after j or j is after i, i.e., the rectangles do not overlap"[11]

In other words it places rectangles in a way that no one overlaps another one. Referring to the resources of the planner, every needed resource of a requirement can be seen as a rectangle. One edge presents the duration the resources are bounded and the other edge represents the number of required resources. Listing 4.9 shows how it is modelled in the proposed release planner. The second argument of *Diff2* represents the start point of the

[11] http://jacop.osolpro.com/

Devel:	Diff2(store,
x_{start}	{ Req1→start,Req2→start,Req3→start,Req4→start, Req5→start,Req6→start,Req7→start},
y_{start}	{ Req1→bounded_developers,Req2→bounded_developers,Req3→bounded_developers,Req4→bounded_developers, Req5→bounded_developers,Req6→bounded_developers,Req7→bounded_developers},
x_{length}	{ Req1→duration,Req2→duration,Req3→duration,Req4→duration, Req5→duration,Req6→duration,Req7→duration},
y_{length}	{ Req1→needed_developers ,Req2→needed_developers,Req3→needed_developers,Req4→needed_developers, Req5→needed_developers,Req6→needed_developers,Req7→needed_developers},
Testers:	Diff2(store,
x_{start}	{ Req1→start,Req2→start,Req3→start,Req4→start, Req5→start,Req6→start,Req7→start},
y_{start}	{ Req1→bounded_testers,Req2→bounded_testers,Req3→bounded_testers,Req4→bounded_testers, Req5→bounded_testers,Req6→bounded_testers,Req7→bounded_testers},
x_{length}	{ Req1→duration,Req2→duration,Req3→duration,Req4→duration, Req5→duration,Req6→duration,Req7→duration},
y_{length}	{ Req1→needed_testers,Req2→needed_testers,Req3→needed_testers,Req4→needed_testers, Req5→needed_testers,Req6→needed_testers,Req7→needed_testers},
Budget:	Diff2(store,
x_{start}	{ Req1→release_start,Req2→release_start,Req3→release_start,Req4→release_start, Req5→release_start,Req6→release_start,Req7→release_start},
y_{start}	{ Req1→bounded_budget,Req2→bounded_budget,Req3→bounded_budget,Req4→bounded_budget, Req5→bounded_budget,Req6→bounded_budget,Req7→bounded_budget},
x_{length}	{ Req1→release_duration,Req2→release_duration,Req3→release_duration,Req4→release_duration, Req5→release_duration,Req6→release_duration,Req7→release_duration},
y_{length}	{ Req1→needed_budget,Req2→needed_budget,Req3→needed_budget,Req4→needed_budget, Req5→needed_budget,Req6→needed_budget,Req7→needed_budget},

Table 4.16: Diff2 constraints for the resources developers, testers, and budget.

x-axis of the rectangles. In the cases of developers and testers the second argument has to be the start day of the requirement implementations, for the budget it has to be the start day of the releases to which the requirements are assigned. The reason for this difference is easy to explain. Developers and testers are available again after they have completed a requirement implementation, so the start point is equal to the implementation start. This is not true for money, it can not be reused after it has been spent. Therefore, used budget is bounded to the whole release. The third argument presents the start point of the y-axis. In our case this is simply the amount of resources that are already in use. The fourth and fifth arguments take the length of each dimension. For the first dimension it means that developers and testers are bounded to the duration of the requirement implementation and the budget is bounded to the whole release. The length of the second dimension is given by the number of required resources. In Figure 4.5 the rectangles, based on the presented solution of Section 4.4.1 are illustrated. The created *Diff2* constraints do not consider limits, but since the relations between bounded, required, and used resources were defined before, the used resources can now be limited depending on the available resource of the release. Therefore, it is necessary to define a relation between the requirements and the releases to which they are assigned. This is shown in Listing 4.10.

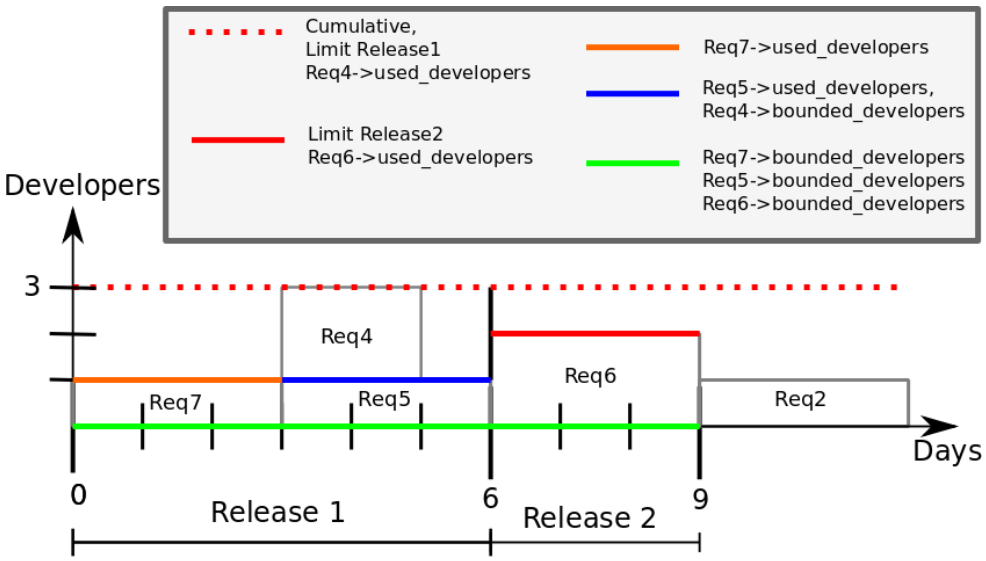


Figure 4.5: Graphical representation of the resource constraints, using the example of developers.

```
1  for(int rel=0; rel<num_releases; rel++)
2  {
3    for(int req=0; req<num_requirements; req++)
4    {
5      release_constraints = new ArrayList<PrimitiveConstraint>();
6      release_constraints.add(new XlteqC(used_developers[req],num_devel_release));
7      release_constraints.add(new XlteqC(used_testers[req],num_tester_release));
8      release_constraints.add(new XlteqC(used_budget[req],budget_release));
9      release_constraints.add(new XeqY(release_durations_of_requirements[req],
           release_durations[rel]));
10     release_constraints.add(new XeqY(release_start_of_requirements[req],release_starts[
           rel]));
11     release_constraints.add(new XeqC(assigned_releases[req],rel));
12
13     addConstraint(new IfThenElse(new And(new XgteqY(impl_starts[req],release_starts[rel])
           ,new XlteqY(impl_ends[req],release_ends[rel])),
14               new And(release_constraints),
15               new XneqC(assigned_releases[j],i)));
16   }
17 }
```

Listing 4.10: Requirement implementation constraints, depending on the assigned release.

In lines 5-11 all constraints are defined that have to be considered if the requirement is assigned to a particular release. In lines 6, 7, and 8 the resources are limited to the maximum of a release. Based on the relations between bounded, needed, and used resources and the usage of theses FDVs in *Diff2*, it is possible to limit the resources of a particular release. It was mentioned before, that the resource budget is not bounded for the duration

of a requirement implementation, but for the whole release. Therefore, the lines 9 and 10 store the start time and duration of the release to which a requirement is scheduled. Line 11 stores the release for every requirement to which it is assigned to an own FDV. This seems to be unnecessary, because the assigned release can be concluded from the start time of a requirement. Later in this section, especially in Section 4.4.4 it is shown that these assignments are helpful to improve the search performance. As it can be seen in Listing 4.10, these constraints are generated for every release. However, a requirement can only be assigned to one release and so only the constraints referring to this release shall be considered. This can be ensured by the conditional constraint *IfThenElse*. Since this constraint is generated for every requirement and for every release, in the presented example 14 *IfThenElse* constraints are generated, see Table 4.17.

Release 1 "Authentication":	IfThenElse(And(XgteqY(Req1→start,Rel1→start),XlteqY(Req1→end,Rel1→end)),	Line 13
	And(XlteqC(Req1→used_developers,3), XlteqC(Req1→used_testers,2), XlteqC(Req1→used_budget,500), XeqY(Req1→release_duration,Rel1→duration), XeqY(Req1→release_start,Rel1→start), XeqC(Req1→assigned_release, 0))	Line 6-11
	XneqC(Req1→assigned_release, 0))	Line 15
Release 1 "Apache Authentication":	IfThenElse(And(XgteqY(Req2→start,Rel1→start),XlteqY(Req2→end,Rel1→end)),	Line 13
	And(XlteqC(Req2→used_developers,3), XlteqC(Req2→used_testers,2), XlteqC(Req2→used_budget,500), XeqY(Req2→release_duration,Rel1→duration), XeqY(Req2→release_start,Rel1→start), XeqC(Req2→assigned_release, 0))	Line 6-11
	XneqC(Req2→assigned_release, 0))	Line 15
Release 1 "Database Authentication":	IfThenElse(And(XgteqY(Req3→start,Rel1→start),XlteqY(Req3→end,Rel1→end)),	Line 13
	And(XlteqC(Req3→used_developers,3), XlteqC(Req3→used_testers,2), XlteqC(Req3→used_budget,500), XeqY(Req3→release_duration,Rel1→duration), XeqY(Req3→release_start,Rel1→start), XeqC(Req3→assigned_release, 0))	Line 6-11
	XneqC(Req3→assigned_release, 0))	Line 15
Release 1 "Auth. Tables":	IfThenElse(And(XgteqY(Req4→start,Rel1→start),XlteqY(Req4→end,Rel1→end)),	Line 13
	And(XlteqC(Req4→used_developers,3), XlteqC(Req4→used_testers,2), XlteqC(Req4→used_budget,500), XeqY(Req4→release_duration,Rel1→duration), XeqY(Req4→release_start,Rel1→start), XeqC(Req4→assigned_release, 0))	Line 6-11
	XneqC(Req4→assigned_release, 0))	Line 15
Release 1 "Auth. DB Communication":	IfThenElse(And(XgteqY(Req5→start,Rel1→start),XlteqY(Req5→end,Rel1→end)),	Line 13
	And(XlteqC(Req5→used_developers,3), XlteqC(Req5→used_testers,2), XlteqC(Req5→used_budget,500), XeqY(Req5→release_duration,Rel1→duration), XeqY(Req5→release_start,Rel1→start), XeqC(Req5→assigned_release, 0))	Line 6-11
	XneqC(Req5→assigned_release, 0))	Line 15
Release 1 "Auth. Password Encryption":	IfThenElse(And(XgteqY(Req6→start,Rel1→start),XlteqY(Req6→end,Rel1→end)),	Line 13
	And(XlteqC(Req6→used_developers,3), XlteqC(Req6→used_testers,2), XlteqC(Req6→used_budget,500), XeqY(Req6→release_duration,Rel1→duration), XeqY(Req6→release_start,Rel1→start), XeqC(Req6→assigned_release, 0))	Line 6-11
	XneqC(Req6→assigned_release, 0))	Line 15

Release 1 “Database Server”:	IfThenElse(And(XgteqY(Req7→start,Rel1→start),XlteqY(Req7→end,Rel1→end)),	Line 13
	And(XlteqC(Req7→used_developers,3), XlteqC(Req7→used_testers,2), XlteqC(Req7→used_budget,500), XeqY(Req7→release_duration,Rel1→duration), XeqY(Req7→release_start,Rel1→start), XeqC(Req7→assigned_release, 0))	Line 6-11
	XneqC(Req7→assigned_release, 0))	Line 15
Release 2 “Authentication”:	IfThenElse(And(XgteqY(Req1→start,Rel2→start),XlteqY(Req1→end,Rel2→end)),	Line 13
	And(XlteqC(Req1→used_developers,2), XlteqC(Req1→used_testers,1), XlteqC(Req1→used_budget,0), XeqY(Req1→release_duration,Rel2→duration), XeqY(Req1→release_start,Rel2→start), XeqC(Req1→assigned_release, 1))	Line 14/6-11
	XneqC(Req1→assigned_release, 1))	Line 15
Release 2 “Apache Authentication”:	IfThenElse(And(XgteqY(Req2→start,Rel2→start),XlteqY(Req2→end,Rel2→end)),	Line 13
	And(XlteqC(Req2→used_developers,2), XlteqC(Req2→used_testers,1), XlteqC(Req2→used_budget,0), XeqY(Req2→release_duration,Rel2→duration), XeqY(Req2→release_start,Rel2→start), XeqC(Req2→assigned_release, 1))	Line 14/6-11
	XneqC(Req2→assigned_release, 1))	Line 15
Release 2 “Database Authentication”:	IfThenElse(And(XgteqY(Req3→start,Rel2→start),XlteqY(Req3→end,Rel2→end)),	Line 13
	And(XlteqC(Req3→used_developers,2), XlteqC(Req3→used_testers,1), XlteqC(Req3→used_budget,0), XeqY(Req3→release_duration,Rel2→duration), XeqY(Req3→release_start,Rel2→start), XeqC(Req3→assigned_release, 1))	Line 14/6-11
	XneqC(Req3→assigned_release, 1))	Line 15
Release 2 “Auth. Tables”:	IfThenElse(And(XgteqY(Req4→start,Rel2→start),XlteqY(Req4→end,Rel2→end)),	Line 13
	And(XlteqC(Req4→used_developers,2), XlteqC(Req4→used_testers,1), XlteqC(Req4→used_budget,0), XeqY(Req4→release_duration,Rel2→duration), usedXeqY(Req4→release_start,Rel2→start), XeqC(Req4→assigned_release, 1))	Line 14/6-11
	XneqC(Req4→assigned_release, 1))	Line 15
Release 2 “Auth. DB Communication”:	IfThenElse(And(XgteqY(Req5→start,Rel2→start),XlteqY(Req5→end,Rel2→end)),	Line 13
	And(XlteqC(Req5→used_developers,2), XlteqC(Req5→used_testers,1), XlteqC(Req5→used_budget,0), XeqY(Req5→release_duration,Rel2→duration), XeqY(Req5→release_start,Rel2→start), XeqC(Req5→assigned_release, 1))	Line 14/6-11
	XneqC(Req5→assigned_release, 1))	Line 15
Release 2 “Auth. Password Encryption”:	IfThenElse(And(XgteqY(Req6→start,Rel2→start),XlteqY(Req6→end,Rel2→end)),	Line 13
	And(XlteqC(Req6→used_developers,2), XlteqC(Req6→used_testers,1), XlteqC(Req6→used_budget,0), XeqY(Req6→duration,Rel2→duration), XeqY(Req6→release_start,Rel2→start), XeqC(Req6→assigned_release, 1))	Line 14/6-11
	XneqC(Req6→assigned_release, 1))	Line 15
Release 1 “Database Server”:	IfThenElse(And(XgteqY(Req7→start,Rel2→start),XlteqY(Req7→end,Rel2→end)),	Line 13
	And(XlteqC(Req7→used_developers,2), XlteqC(Req7→used_testers,1), XlteqC(Req7→used_budget,0), XeqY(Req7→duration,Rel2→duration), XeqY(Req7→release_start,Rel2→start), XeqC(Req7→assigned_release, 1))	Line 14/6-11
	XneqC(Req7→assigned_release, 1))	Line 15

Table 4.17: Requirement constraints depending on the assigned release of the presented example.

Another requirement of the prototype is met in Listing 4.11. Since not only the amount of developers and testers can be defined for a requirement, but also specific persons can be assigned to particular requirements, it has to be ensured that requirement implementations that need the same person, do not overlap. Table 4.18 shows such a constraint, if the requirements "Database Server" and "Auth. DB Communication" shall be implemented by the same developer.

```
1 private void addSameResourceConstraint(int req1, int req2)
2 {
3   new Or(new XlteqY(impl_ends[req1],impl_starts[req2]),
4              new XgteqY(impl_starts[req1],impl_ends[req2])));
5 }
```

Listing 4.11: The same resource for two requirements

"Database Server" & "Auth. DB Communication" implemented by the same developer:	Or(XlteqY(Req7→end,Req5→start), XgteqY(Req7→start,Req5→end))

Table 4.18: Req7 and Req5 cannot be implemented in parallel, because they require the same resource.

Requirement Relations Constraints

In Section 4.2 various relations between requirements are presented. The prototype implements four relations that basically follow the definitions of Goknil et al. [2010]. In addition to the problem of selecting the most relevant requirement relations, the mix-up of relation names requires to choice a formalization. It was decided to use names that are well-known in the area of configuration systems and are also used in the unified modelling language (UML) [Felfernig et al., 2000]. Furthermore, a detailed description of the relations, impacts on the release plans, and the implementations of these relations with JaCoP are presented. In Figure 4.6 an overview about the relations, defined for the example is given. For a better understanding of the presented constraints, all generated ones based on this model are shown and described.

Part-refinement This relation allows to split a requirement into small parts. In this work the term feature was already discussed in Section 3.3.1. This relation allows to group requirements to a logical unit and therefore it can be used to model features. This relation can be seen as a composition, known from UML. The *part-refinement* and the *generalization* relations require to introduce a specific requirement type. In Table 4.1 Req1 "Authentication" and Req3 "Database Authentication" are marked as abstract and no resources are

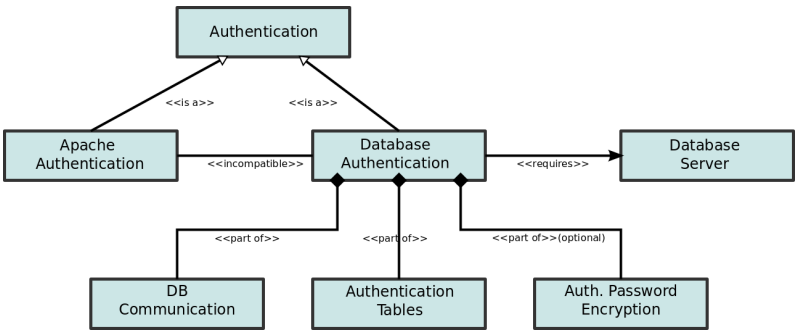


Figure 4.6: Relations between requirements of the presented example.

defined for them. Indeed, abstract requirements do not need any kind of resources. This does not mean that they can be scheduled randomly. Their scheduling is influenced by their relations to other requirements. In the case of the *part-refinement* relation an abstract requirement is completed if all its sub-requirements are completed.
The three requirements "Table Design", "DB Communication", and "Password Encryption" can be combined to one abstract requirement "Database Authentication", see Figure 4.7. This means that the requirement "Database Authentication" is completed if all three sub-requirements are completed. Later in this section it can be seen that all constraints of the abstract requirement are also valid for the sub-requirements. It shall also be mentioned that this kind of relation is often called *contains*. Actually, this depends on the point of view. The prototype supports a *part of* relation to assign a requirement to another one. For a better overview, the browsing window of the prototype provides decomposing an abstract requirement that contains sub-requirements. Figure 4.7 (a), (b), (c) shows how to model the above example, (d) illustrates how it is displayed in the browsing window.

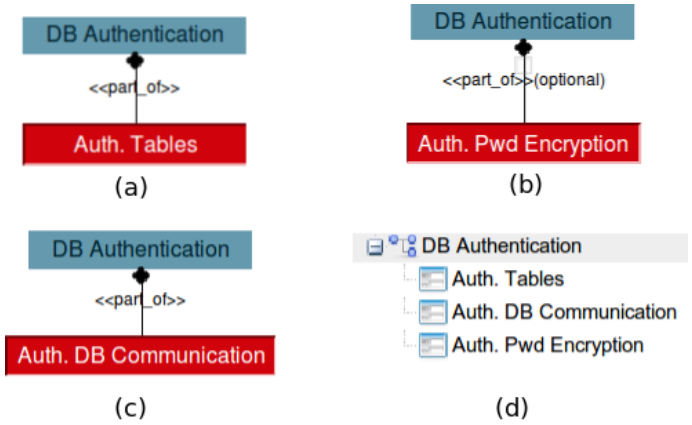


Figure 4.7: Prototype representation of *part-refinement* relations.

In Figure 4.7 (b) a specific use of the *part of* relation is shown. Generally a requirement which is decomposed into sub parts is fulfilled if all the sub-requirements are completed. This is not absolutely necessary for sub-requirements that are declared as *optional*. Referring to the presented example, the requirement "Database Authentication" is completed as soon as "DB Communication" and "Table Design" are fulfilled. This is especially useful for other requirements that need "Database Authentication". They can be implemented before "Password Encryption" is completed. In the lines 5-12 of Listing 4.12 the end times and the assigned releases of the mandatory sub-requirements are stored in arrays. Finally in line 15 the *Max* constraint ensures that the abstract requirement is completed when the last mandatory sub-requirement implementation has been finished. In line 16 the same constraint is imposed in a weakened form. In contrast to the first constraint it only expresses that the abstract requirement is completed in the same release as the last sub-requirement implementation. As mentioned before this will improve the searching performance, see Section 4.4.4. In Table 4.19 it can be seen that "Database Authentication" (Req3) is finished when all mandatory requirements (Req4, Req5) have been implemented.

```
1  for(int req = 0; req < num_requirements; req++)
2  {
3    ArrayList<FDV> parts = new ArrayList<FDV>();
4    ArrayList<FDV> part_releases = new ArrayList<FDV>();
5    for(Integer part_id : part_ids)
6    {
7      if(!requirement_store.get(part_id).isOptional(requirement.getId()))
8      {
9        parts.add(impl_ends[requirement_index.get(part_id)]);
10       part_releases.add(assigned_releases[requirement_index.get(part_id)]);
11     }
12   }
13   if(!parts.isEmpty())
14   {
15     addConstraint(new Max(impl_ends[req],parts));
16     addConstraint(new Max(assigned_releases[req],part_releases));
17   }
18 }
```

Listing 4.12: Modelling *part-refinement* relations with JaCop.

"DB Authentication" is done when "Auth. Tables" and "Auth. DB Communication" are implemented.	Max(Req3→end, {Req4→end,Req5→end})
Assigned Release of "DB Authentication"	Max(Req3→assigned_release, {Req4→assigned_release,Req5→assigned_release}))

Table 4.19: Defining the end of "DB Authentication" based on its sub-requirements.

For practical use of this relation the prototype provides the possibility to schedule the implementations of sub-requirements close together. This is done by defining a maximum range between the start of the first sub-requirement implementation and the end of the last sub-requirement implementation. This is done by the code of Listing 4.13. In lines 9-18 the start time and end time FDVs and the total duration of the sub-requirements are stored to local variables. In lines 21-23 some auxiliary FDVs are defined to store the start time of the first sub-requirement implementation, the actual end time, and the maximum allowed end time of the last sub-requirement implementation. Line 27 defines that the last sub-requirement has to be finished at least after the total durations of all sub-requirements. In other words it means that when the sub-requirements can not be implemented in parallel they have at least to be done in a row.

```
1  for(int req = 0; req<num_requirements_; req++)
2  {
3    int parent_id = requirement_ids[req];
4    ArrayList<FDV> mandatory_parts_starts = new ArrayList<FDV>();
5    ArrayList<FDV> mandatory_parts_ends = new ArrayList<FDV>();
6    ArrayList<Integer> mandatory_part_ids = new ArrayList<Integer>();
7    //total duration of all parts
8    int bundled_duration = 0;
9    for(Integer part_id : requirement_store.getRelationFinder().getParts(parent_id))
10   {
11     AbstractRequirement part = requirement_store_.get(part_id);
12     if(!part.isOptional(parent_id))
13     {
14       mandatory_parts_starts.add(impl_starts[requirement_index.get(part_id)]);
15       mandatory_parts_ends.add(impl_ends[requirement_index.get(part_id)]);
16       bundled_duration+=part.getDuration();
17     }
18   }
19   if(mandatory_parts_starts.size() > 0)
20   {
21     FDV bundle_start=new FDV(store_,"Bundle start"+requirement.getName(),0,total_requirement_duration);
22     FDV bundle_end=new FDV(store_,"Bundle end "+requirement.getName(),0,total_requirement_duration);
23     FDV latest_bundle_end=new FDV(store_,"Max. bundle end"+requirement.getName(),0,
           total_requirement_duration);
24     addConstraint(new XplusCeqZ(bundle_start,bundled_duration,latest_bundle_end));
25     addConstraint(new Min(bundle_start,mandatory_parts_starts));
26     addConstraint(new Max(bundle_end,mandatory_parts_ends));
27     addConstraint(new XgteqY(latest_bundle_end, bundle_end));
28     for(int j = 1; j < mandatory_part_ids.size();j++)
29       addConstraint(new XeqY(assigned_releases[requirement_index.get(mandatory_part_ids.get(j))],
30                     assigned_releasess[requirement_index.get(mandatory_part_ids.get(j-1))]));
31
32   }
33 }
```

Listing 4.13: Bundling of sub-requirements with JaCoP.

Start of “DB Authentication”	Min(bundle_start,Req4→start,Req5→start)
End of “DB Authentication”	Max(bundle_end,Req4→end,Req5→end)
Latest end of “DB Authentication”	XplusCeqZ(bundle_start,5,latest_bundle_end)
Bundle “DB Authentication”	XgteqY(lastest_bundle_end,bundle_end)
“DB Authentication” in one release	XeqY(Req4→assigned_release,Req5→assigned_release)

Table 4.20: Bundling the sub-requirements of “DB Authentication”.

The first constraint of Table 4.20 defines that “DB Authentication” starts when one of the sub-requirement implementations “Auth. Tables” (Req4) or “Auth. DB Communication” (Req5) start. The second one determines the end of “DB Authentication”. The third constraint ensures that “DB Authentication” has to be finished within 5 days, because that is the duration it would take if “Auth. Tables” and “Auth. DB Communication” are implemented in succession. Finally, the constraint is imposed to actually ensure that “DB Authentication” is completed within 5 days. The last constraint ensures that “DB Authentication” is implemented in one release.

Generalization In the domain of configuration, generalization means that attributes, ports, and constraints are inherited from one type to a subtype [Felfernig et al., 2000]. Goknil et al. [2010] emphasize a relation *refines* that also inherits the properties of a requirement to a sub-requirement. They define that a refinement is an extension of an existing requirement. In contrast to these definitions, general requirements have to be defined as abstract in the proposed prototype. Therefore, they are more comparable to the relation between an abstract class to derived concrete classes, known from object-oriented programming. Similarly to the *part-refinement* relation, the scheduling depends on the derived, concrete requirements. In generalization a requirement is fulfilled as soon as one of its derived requirements is implemented. The implementation of generalization, see Listing 4.14, is quite similar to the *part-refinement* relation, see Listing 4.12, but instead of using the *Max* constraint, the *Min* constraint is used. Table 4.21 shows the generated constraints in the presented example.

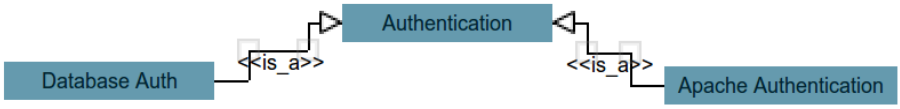


Figure 4.8: Prototype representation of *generalization* relations.

In the presented example “Authentication” (Req1) is a generalization of the requirements “Apache Authentication” (Req2) and “Database Authentication” (Req3). This is illustrated in Figure 4.8. This relation is labeled as *is a*, which is inspired by UML and ERM. This kind of relation is often useful to model alternatives. Later in this section the two refinements “Apache Authentication” and “Database Authentication” are defined to be incompatible. This means that only one of them shall be implemented. Since it is not known which of them will be in a release plan, relations can now be connected to the abstract requirement. This is very useful for requirements that need a kind of authentication, but do not care

about details. In Figure 4.8 “Apache Authentication” and “Database Authentication” are not defined to be incompatible to each other. In contrast to the generalization described by Felfernig et al. [2000], here derived requirements are not disjunctive by default.

```
1  for(int req = 0; req < num_requirements; req++) {
2    ArrayList<FDV> child_endtimes = new ArrayList<FDV>();
3    ArrayList<FDV> child_releases = new ArrayList<FDV>();
4    for(Integer child_id : requirement_store.getRelationFinder().getChilds(requirement_ids[
         req]))
5    {
6      child_endtimes.add(impl_ends[requirement_index.get(child_id)]);
7      child_releases.add(assigned_releases[requirement_index.get(child_id)]);
8    }
9
10   if(!child_endtimes.isEmpty())
11   {
12     addConstraint(new Min(impl_ends[req],child_endtimes));
13     addConstraint(new Min(assigned_releases[req],child_releases));
14   }
15 }
```

Listing 4.14: Modelling *generalization* relations with JaCop.

“Authentication” is done when “Auth. Tables” or “Auth. DB Communication” is implemented.	Min(Req1→end,{Req2→end,Req3→end}))
Release assignment of “Authentication”	Min(Req1→assigned_release, {Req2→assigned_release,Req3→assigned_release}))

Table 4.21: Defining the end of “DB Authentication” based on its refinements.

Requires *Requires* is the most well-defined type of relations among the literature. This relation denotes that a requirement cannot be implemented until all requirements are fulfilled, that are connected to it by the *requires* relation. In the presented example a database server has to be set up before implementing “Database Authentication”. Therefore, it is defined that requirement “Database Authentication” (Req3) requires “Database Server” (Req7). This is illustrated in Figure 4.9. In other words, before database tables can be created and interactions with a database can be implemented, a database server has to be set up. In JaCoP this constraint can simply be modelled by defining that all the needed requirements have to be completed before the particular requirement can start.

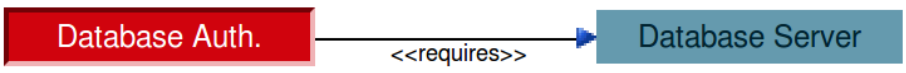


Figure 4.9: Prototype representation of a *requires* relation.

```
1 private void addRequireRelations(int requires_index, ArrayList<Integer>
      required_requirements)
2 {
3   for(Integer required_id : required_requirements)
4   {
5     int required_index = requirement_indexes.get(required_id);
6     addConstraint(new XlteqY(impl_ends[required_index], impl_starts[requires_index]);
7     addConstraint(new XlteqY(assigned_release[required_index], assigned_release[
          requires_index]));
8   }
9  }
```

Listing 4.15: Modelling *requires* relations with JaCop.

The method of Listing 4.15 is called for every requirement that has defined *required* relations. The first argument represents the index to access the start time FDV. The second one holds all requirement Ids that are required by the current requirement. The constraint in line 6 ensures that all needed requirement implementations have already been finished before the current one can start. In line 7 it is ensured that the needed requirements are implemented in an earlier release or at least in the same release. This method is also used to inherit relations from abstract requirements to their refinements or their sub-requirements. Therefore, this method is simply invoked with the index of the sub-requirement or the refinement as first argument and the list of all needed requirements of the abstract requirement. This is done for all refinements or all sub-requirements. The first two constraints in Table 4.22 reflect the *requires* relation introduced in Figure 4.9. The remaining constraints are generated, because of the *part refinement* relations illustrated in Figure 4.7. This shows the advantages of the above presented *part refinement* relation, see Section 4.4.3. Instead of defining the *requires* relations for all sub-requirements it is sufficient to define it for the requirement "DB Authentication".

"DB Authentication" requires "Database Server"	XlteqY(Req7→end, Req3→start) XlteqY(Req7→assigned_release, Req3→assigned_release)
"Auth. Tables" requires "Database Server"	XlteqY(Req7→end, Req4→start) XlteqY(Req7→assigned_release, Req4→assigned_release)
"Auth. DB Communication" requires "Database Server"	XlteqY(Req7→end, Req5→start) XlteqY(Req7→assigned_release, Req5→assigned_release)
"Auth. Pwd Encryption" requires "Database Server"	XlteqY(Req7→end, Req6→start) XlteqY(Req7→assigned_release, Req6→assigned_release)

Table 4.22: Constraints for the requires relation of "DB Authentication" and its sub-requirements.

Incompatible The *incompatible* relation denotes that only one of two requirements can be implemented. In some literature this type is also called *conflicts* [Goknil et al., 2010; Lin et al., 1996]. Figure 4.10 illustrates this relation for the presented example. It defines that authentication can either be realized by the requirement "Apache Authentication" or the "Database Authentication". This means that only one type of authentication shall be implemented.

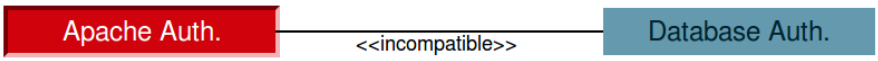


Figure 4.10: Prototype representation of an *incompatible* relation.

```
1 private void addIncompatibleRelations(int index1, int index2)
2 {
3   addConstraint(new IfThen(new XltC(assigned_releases[index1],num_releases),
4                 new XeqC(assigned_releases[index2],num_releases)));
5 }
```

Listing 4.16: Modelling *incompatible* relations with JaCop.

It was already mentioned that requirements that shall not be implemented are scheduled after the project end, respectively to the last release. In Listing 4.16 it can be seen seen that only one *IfThen* constraint is necessary to model an *incompatible* relation. This relation is inherited to all sub-requirements and all refinements. Therefore, the proposed constraint is also generated for "Auth. Tables" (Req4), "Auth. DB Communication" (Req5), and "Auth. Password Encryption" (Req6), see Table 4.23. The first constraint below expresses that either "Apache Authentication" or "Database Authentication" has to be implemented. The last constraints ensure that if "Apache Authentication" shall be implemented, the sub-requirements of "Database Authentication" must not be implemented. Since 0 and 1 represent the actual releases of the presented example, all rejected requirements are scheduled to 2.

4.4.4 Searching

This section shows how a solution can be found based on the variables and constraints defined before. In Section 4.4.1 it was defined that a solution consists of the start and end times of each release and of each requirement implementation. Hence, it is obvious to search an assignment to the FDVs defined in Table 4.4. As expected, this search method provides a lot of release plans. However, having a closer look to these results it becomes clear that such a search generates a lot of equal release plans. Such a solution is shown in Table 4.24.

"Apache Authentication" is incompatible to "Database Authentication"	IfThen(XltC(Req2→assigned_release,2), XeqC(Req3→assigned_release,2))
"Apache Authentication" is incompatible to "Auth. Tables"	IfThen(XltC(Req2→assigned_release,2), XeqC(Req4→assigned_release,2))
"Apache Authentication" is incompatible to "Auth. DB Communication"	IfThen(XltC(Req2→assigned_release,2), XeqC(Req5→assigned_release,2))
"Apache Authentication" is incompatible to "Auth. Password Encryption"	IfThen(XltC(Req2→assigned_release,2), XeqC(Req6→assigned_release,2))

Table 4.23: Constraints to express the incompatibility between "DB Authentication" and its sub-requirements to "Apache Authentication".

The implementation date of "Database Server" (Req7) and "Authentication Tables" (Req4) in Table 4.24 are swapped and therefore these solutions differ. However, from the point of the stakeholders these solutions are equal, because both requirements are available after the first release. The attentive reader might have noticed that the second solution of Table 4.24 is not valid in the presented example because of the *requires* relation between "DB Authentication" (Req3) and "Database Server" (Req7). For demonstration purposes this relation was ignored. Because of the small release ranges, the few requirements, and the number of relations between the requirements in the presented example, the disadvantages of this search method does not have consequences. However, in larger requirements models a large number of redundant release plans would be generated. This behaviour can be avoided by dividing the search into a master search and a slave one. The master search finds different assignments of requirements to releases, while the slave search finds the actual solution. Listing 4.17 shows how a search can be done with JaCoP. Lines 1 to 11 define for which variables assignments are searched. The master search shall only search for different release plans, considering the above explanation. Line 10 is the reason that some constraints are not only defined for the start and end times of the requirement implementations, but also for release assignments. This improves the quality of the master search, significantly. Therefore, it should be mentioned that not for all solutions, found by the master search, a solution can be found by the slave search. Adding constraints that influence the master search reduces such wrong solutions.

Rel1	Rel2	Req1	Req2	Req3	Req4	Req5	Req6	Req7
0 6	6 9	6 6	9 12	6 6	3 5	3 6	6 9	0 3
0 6	6 9	6 6	9 12	6 6	0 2	3 6	6 9	3 6

Table 4.24: Identical release plans, but different solutions.

```
1  for(int rel = 0; rel < num_releases_; rel++)
2  {
3    slave_vars_.add(release_starts_[rel]);
4    slave_vars_.add(release_ends_[rel]);
5  }
6  for(int req = 0; req < num_requirements_; req++)
7  {
8    slave_vars_.add(impl_starts_[req]);
9    slave_vars_.add(impl_ends_[req]);
10   master_vars_.add(assigned_releases[req]);
11 }
12 SelectChoicePoint master_select = new SimpleSelect(master_vars_.toArray(new Variable[
       master_vars_.size()]), new SmallestMin(),new SmallestDomain(), new IndomainMin());
13 master_search_ = new DepthFirstSearch();
14 master_search_ .getSolutionListener().searchAll(true);
15 slave_search_ = new DepthFirstSearch();
16 SelectChoicePoint slave_select = new SimpleSelect(slave_vars_.toArray(new Variable[
       slave_vars_.size()]), new SmallestMin(),new SmallestDomain(), new IndomainMin());
17 slave_search_.setSelectChoicePoint(slave_select);
18 master_search_.addChildSearch(slave_search_);
19 result = master_search_ .labeling(store_, master_select);
```

Listing 4.17: Searching for solutions.

In lines 12 and 16 the *SelectChoicePoint* instances for the master and slave search are created. This class defines how the search algorithm shall work, including the selection of the variables and the values. The proposed prototype uses three different selection types:

- *SmallestMin* selects the FDV with the smallest value in its domain.
- *SmallestDomain* selects FDV which has the smallest domain size.
- *IndomainMin* selects a minimal value from the current domain of FDV.

In Listing 4.17, the lines 13 and 15 show that a depth-first-search algorithm is used to find solutions. Line 14 defines that a search shall not stop after a solution has been found. This does not mean that every run has to find all solutions. It is possible to define a timeout or a limit of solutions. Before starting a search the slave search has to be added to the master, this is done in line 18. Finally a search starts by invoking the *labeling* method of the master search. The created solutions are presented as an array of arrays. Every array presents one solution in the form that was already introduced in Section 4.4.1. Table 4.25 shows two solutions that could be found for the presented example.

In Figure 4.11 the first release plan (a) and the second one (b) are displayed by the proposed prototype. It can be seen that in both cases the authentication is done with a database. Since no release has enough budget to realize “Apache Authentication”, it can not be implemented. The only difference between the two plans is that “Auth. Password Encryption” shall be implemented in plan 1, but not in plan 2. This is possible, because

this requirement was marked as an optional sub-requirement of “Database Authentication”, see Section 4.4.3. It can also be seen that requirements that shall not be implemented are scheduled after the end of the second release and therefore they are not displayed in Figure 4.11. It should also be mentioned that no more solutions are available for the presented example. Because of the relations between the requirements and the constraints of the releases, no further release plans can be found.

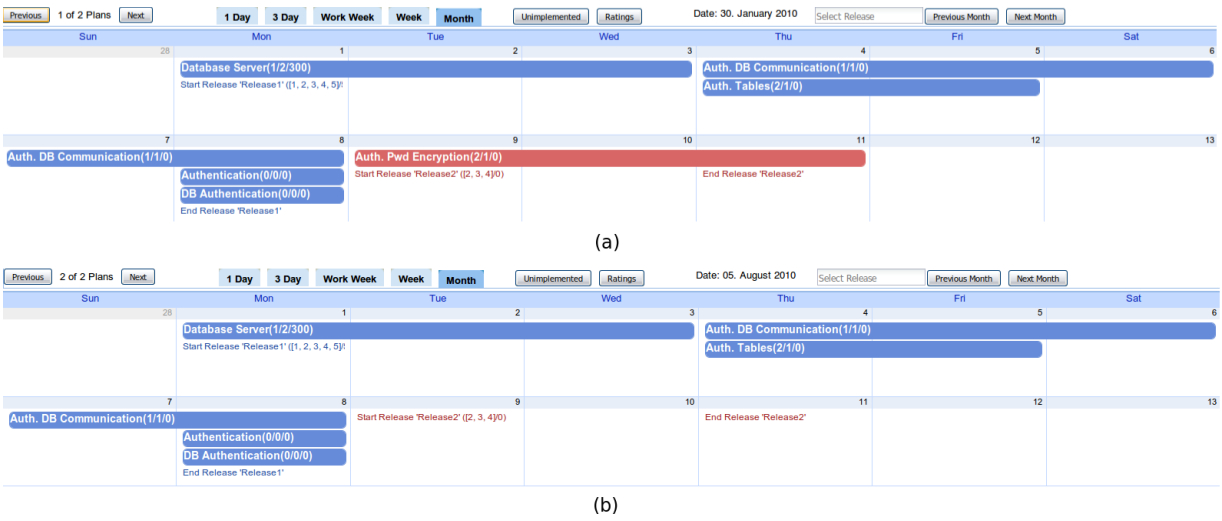


Figure 4.11: Solutions for the minimal example.

Rel1	Rel2	Req1	Req2	Req3	Req4	Req5	Req6	Req7
0 6	6 9	6 6	9 12	6 6	3 5	3 6	6 9	0 3
0 6	6 9	6 6	12 15	6 6	3 5	3 6	9 12	0 3

Table 4.25: Solutions for the presented example.

4.5 Diagnosing Inconsistent Constraints

While using the prototype, situations were identified where no solution that satisfied all constraints could be found. In this situations a diagnosis helps to identify inconsistencies. Therefore, an existing diagnosis component is integrated into the prototype. The foundation for this framework was developed by Felfernig et al. [2004] for diagnosing configuration knowledge bases. This diagnosis component is used to identify inconsistencies in a requirements model. In the case of release planning such inconsistencies can occur because of conflicting relations between requirements. The component identifies the constraints that are responsible for the inconsistencies. The framework uses the QuickXplain algorithm [Junker, 2004] to calculate minimal conflict sets and the Hitting Set algorithm, proposed by Reiter [1987], to identify minimal diagnoses. Based on these diagnoses a minimal set of requirements that have to be adopted in order to find at least one release plan is proposed to the user. It is important to find a minimal set of requirements that have to be changed, because the origin requirements model shall be kept as much as possible. These constraints are identified by the Hitting Set Directed Acyclic Graph (HSDAG) algorithm [Reiter, 1987] and is called diagnosis. This means that a diagnosis identifies the smallest set of constraints that are faulty. In other words in release planning a diagnosis

> is a set $\Delta \subseteq CONST$ s.t. a release plan can be found. A diagnosis $\Delta CONST$ is minimal if there does not exists a diagnosis Δ' with $\Delta' \subset \Delta$.

Furthermore, it ensures that all unidentified constraints are correct and therefore the system is consistent. As mentioned, HSDAG requires the identification of minimal conflict sets (CS). In the case of a release plan

> a conflict set CS is a subset $const_1, const_2, \ldots, const_q \subseteq CONST$, s.t. $CS \cup CONST \cup R$ does not allow the calculation of a release plan. A conflict set CS is minimal iff there does not exist a conflict set CS' with CS' $\subset$ CS.

In the following, the functionality of the HSDAG algorithm shall be described by an example. In Figure 4.12 an inconsistent requirements model is illustrated. It can be seen that Req1 can not be implemented, because the needed requirements Req2 and Req3 are incompatible. The illustrated model creates the following, simplified constraints:

- $const_1$: include *Req2*$\wedge$ include *Req1*
- $const_2$: include *Req3*$\wedge$ include *Req1*
- $const_3$: include *Req2*$\wedge$ exclude *Req3*
- $const_4$: include *Req3*$\wedge$ exclude *Req2*

- $const_5$: include *Req5*∧ include *Req4*
- $const_6$: include *Req5*∧ include *Req6*

In this example three conflict sets can be identified $CS_1\{const_1, const_4\}$, $CS_2\{const_2, const_3\}$, and $CS_3\{const_3, const_4\}$. All three are conflict sets since $CS_1 \cup CONST$, $CS_2 \cup CONST$, and $CS_3 \cup CONST$ are inconsistent, i.e, no release plan exists. They are also minimal, because there does not exist a subset CS‘ such that $CS‘ \subset CS$ and $CS' \cup CONST$ is inconsistent. The initial point for this

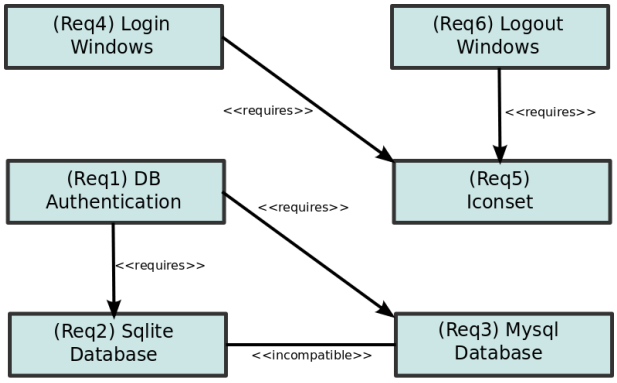


Figure 4.12: Inconsistent requirements model.

example are the presented constraints $\{const_1, const_2, \ldots const_6\}$ and the conflict sets $CS_1\{const_1, const_4\}$, $CS_2\{const_2, const_3\}$, and $CS_3\{const_3, const_4\}$. These sets are generated by the QuickXplain algorithm. Assuming that CS_1 is detected first the corresponded HSDAG looks like illustrated in Figure 4.13. The conflict set CS_1 can be resolved by either removing $const_1$ or $const_4$. The left branch of the tree in Figure 4.13 removes $const_1$ and the right branch removes $const_4$. In both cases the conflict detection algorithm is called, but at this time with the reduced set of constraints. In both cases QuickXPlain returns the conflict set $CS_2 : \{const_2, const_3\}$. This means no diagnosis could be found. Now the procedure can continue by either removing $const_2$ or $const_3$. Removing $const_2$ from the left branch does not find a diagnosis, because another conflict set could be identified, $CS_3 : \{const_3, const_4\}$. The other three branches can not find another conflict set and therefore the diagnoses Δ_1, Δ_2, and Δ_3 have been identified. Continuing on the left branch would identify the diagnoses $\Delta_4 = \{const_1, const_2, const_3\}$ and $\Delta_5 = \{const_1, const_2, const_4\}$. It can be seen that Δ_1 is a subset of Δ_4 and Δ_2 is a subset of Δ_5. This means that Δ_4 and Δ_5 are not minimal. It can also be seen that the diagnoses Δ_1, Δ_2, and Δ_3 are minimal since there does not exist a $\Delta' \subset \Delta$ that fulfills the property of a diagnosis. In summary, HSDAG builds a tree based on the first minimal conflict set, found by the QuickXPlain algorithm. It branches by removing one *const* per

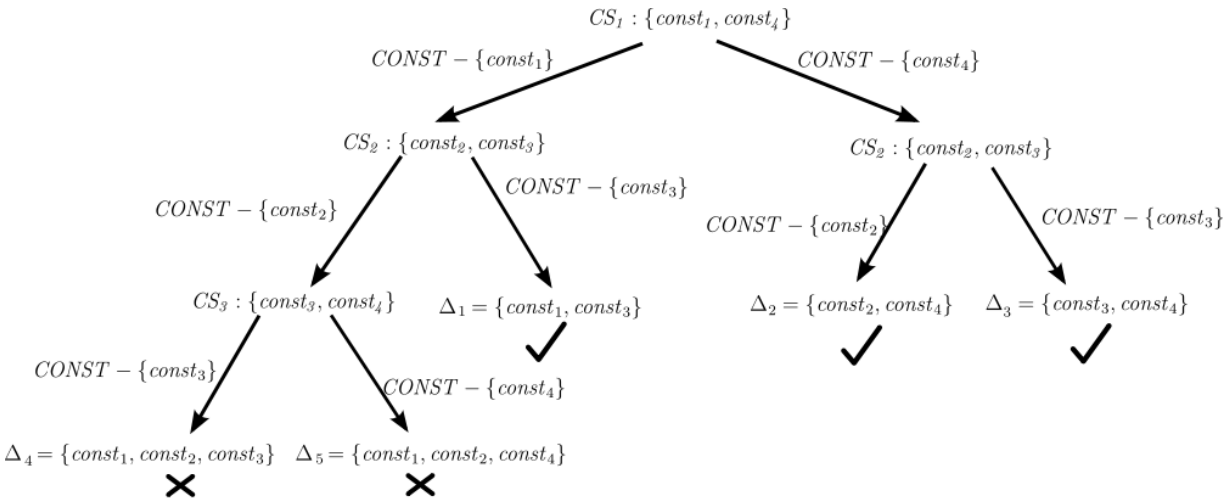


Figure 4.13: Hitting Set Directed Acyclic Graph for an example requirements model.

node of the conflict set from *CONST* as far as a new conflict set can be found. If no more can be identified a diagnosis is found. The algorithm stops after the first diagnosis has been found [Reiter, 1987; Junker, 2004]. Since it searches breadth-first, it ensures that the first diagnosis is minimal. In Section 5.3 the usage of the diagnosis component for the proposed prototype is demonstrated.

4.6 Release Plan Ranking

After the release planner has generated a number of plans, these shall be ranked. Therefore, the users can rate the requirements. In Section 3.3.5 two rating attributes *value* and *urgency* were proposed by Ruhe and Saliu [2005], that are influencing the overall satisfaction of the stakeholders. The rating done by the prototype follows a different approach. An evaluation formula, introduced by Felfernig et al. [2006] for recommender applications is used to rate the generated release plans. In recommender applications a set of products is offered to a user ordered by his interests. Therefore, a user has to rate a number of predefined interest dimensions. In the domain of financial services these may be *profit*, *availability*, and *risk*. Based on the ratings of these dimensions a set of suitable products can be recommended. The recommended products are ordered by Formula 2, where n denotes the number of dimensions, $g(x)$ the utility of a product x, e_i represents the interest of the customer in dimension i, and $s_i(x)$ is the contribution of product x to dimension i. In the domain of release planning, one generated release plan is equivalent to one product. In Section 5.1 it is shown that requirements can be rated by their importance to stakeholders. The

$$g(x) = \sum_{(i=1...n)} e_i s_i(x),$$

Formula 2: Evaluation formula for recommender applications [Felfernig et al., 2006].

user can valuate a requirement by assigning a value between 0 and 10, where 0 means not important and 10 very important. For demonstration purposes the prototype supports rating only by a single person. Of course, in practical applications ratings shall be done by all stakeholders. In Formula 2 different dimensions are used to rate a product. At the moment the presented prototype works with one dimension, *importance*. Since in release planning the products differ from each other by the assignment of requirements to releases, the contribution of a release plan depends on the scheduling. Therefore, Formula 2 has to be adopted. The aim of the ranking is to identify those release plans that implement high rated requirements early and low rated ones later in a project. Hence, the release to which a requirement is assigned can be seen as s. The variable e represents the importance assessed by the user. This has to be added up for all requirements of a release plan. The release plans can now be rated by using Formula 3. In Formula 3, $\#req$ presents the total number of requirements, $\#rel$ the total number of releases, e_j the rating of the user for requirement j, max_rating the highest possible rating, and rel the release to which requirement j is scheduled in the evaluated plan x.

$$f(x) = \frac{\sum_{j=1}^{\#req} \frac{e_j}{max\_rating} * \frac{(\#rel+1-rel)}{\#rel}}{\#req}$$

Formula 3: Evaluation formula for release plans.

In Section 4.4.1 two release plans could be found for the presented requirements model. They only differ since the first one implements “Auth. Password Encryption”, while the second one does not. Table 4.26 shows the ratings of these plans calculated by the proposed formula. As expected, Plan 1 got a higher rating than Plan 2. The best score (1) would be reached if all requirements are rated with value 10 and can be implemented in the first release. In Section 5.2 the proposed formula is used to rank a number of plans generated by the release planner.

Plan	Rating
Plan 1 (with “Auth. Password Encryption”)	0.3929
Plan 2 (without “Auth. Password Encryption”)	0.3571

Table 4.26: Ratings of the example plans.

Chapter 5

Case Study

In this section, the web interface of the implemented prototype is presented. Furthermore, a requirement document of an example application is modelled with the help of the provided interface. The second section describes the generated results and finally shows how the presented diagnosis component (see Section 4.5) is integrated to identify inconsistencies in the model.

5.1 Requirements Model For An Example Application

In Figure 5.1 the main window of the web interface is illustrated. Generally, it is divided into four parts. On the top, a menu bar is available to define new human resources, releases, and requirements. Furthermore, it allows creating release plans, loading existing ones, and opening a window that gives an overview about all requirements and their relations. On the left side, all created objects are listed, which are releases, requirements, and human resources. On the bottom of the main view a logging-area helps to follow the activities and displays error messages. The central part of the view displays the forms to create and edit objects and shows the generated plans. It uses tabs to open more than one form.

Before the requirements model can be created, the releases and their basic data should be defined. Creating a release is quite simple. A release consists of a name, a start range, an end range, and a number of developers and testers. Figure 5.1 shows how a release can be defined. In the illustrated example, the first release starts on 01.03.2010 and ends on 30.04.2010. In Table 5.1 it can be seen that the other two releases do not have fixed start and end dates, but defined ranges. Furthermore, available developers and testers of a release have to be defined. These can be added by double-clicking the desired resource on the left side of the window, see Figure 5.1. Detailed information about all defined releases are listed in Table 5.1.

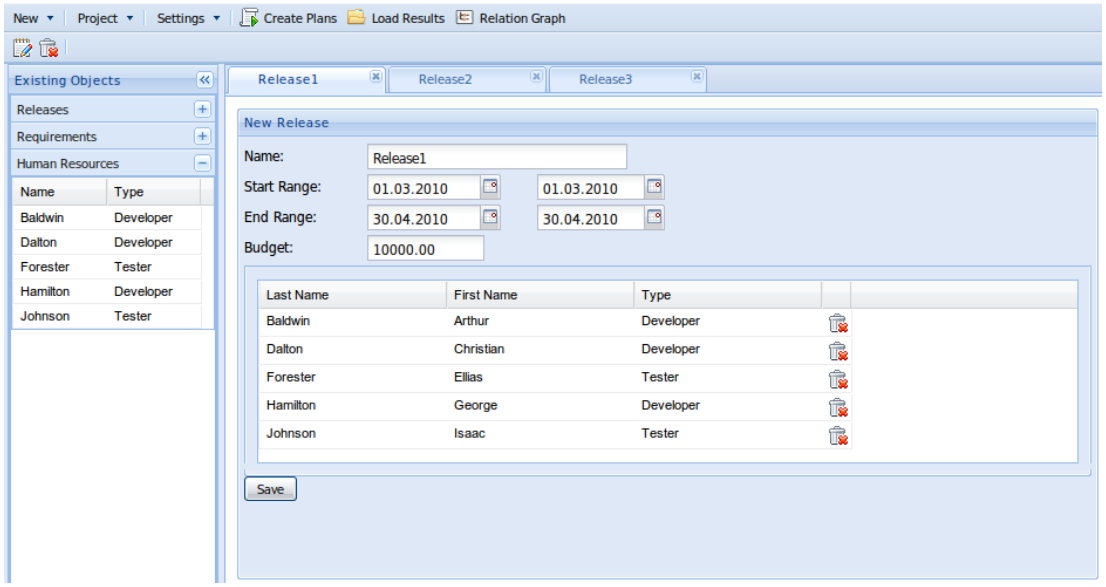


Figure 5.1: Defining a release with the prototype web interface.

Id	Name	Start day	End day	Developers	Testers	Budget
Rel1	Release 1	01.03.2010 - 01.03.2010	30.04.2010 - 30.04.2010	Arthur Baldwin Christian Dalton George Hamilton	Issac Johnson Ellias Forester	10000
Rel2	Release 2	03.05.2010 - 03.05.2010	31.05.2010 - 04.06.2010	Arthur Baldwin Christian Dalton George Hamilton	Issac Johnson Ellias Forester	5000
Rel3	Release 1	01.06.2010 - 07.06.2010	09.07.2010 - 16.07.2010	Arthur Baldwin Christian Dalton	Ellias Forester	5000

Table 5.1: Basic data of the releases of the example application.

Now the requirements have to be defined. The input form for defining requirements consists of three parts. First of all a description and the needed resources of a requirement have to be defined, this is shown in Figure 5.2. There, a name and a description can be defined. The name and the description are later displayed in the generated release plans. The goal describes the primary objective of a requirement in a few words. The dropdown-box “Latest Release” allows to define that a requirement has to be implemented in this or an earlier release, but not later. On the bottom of this form three tabs are available to define the required resources, the relations to other requirements, and to rate a require-

ment. Figure 5.2 shows that the implementation of requirement “Authentication Database Communication” requires one developer, one tester, no additional budget, and it takes 3 days. As mentioned in Section 4.3.1, specific developers and testers can be added to a requirement. In this case the requirement shall be implemented by a developer named “Christian Dalton” and it shall be tested by “Ellias Forester”. The tab “Relations” offers a core element of the proposed prototype, an editor to model relations between requirements. Figure 5.3 (a) shows the relation of requirement “Authentication Database Communication”. It shows that this requirement is a sub-requirement of “Database Authentication”. This editor allows to define all relations presented in Section 4.4.3. Figure 5.4 shows the editor used to model a requirement with some more relations. The last tab of the requirement form provides the possibility to rate a requirement. This is shown in Figure 5.3 (b). As illustrated a requirement can be rated from 0 (unimportant) to 10 (very important). In this example it is rated to 6. In Section 4.6 it was already described how these ratings influence the order of the generated release plans. Table 5.2 shows the basic data of the requirements, created for the example application and in Figure 5.5 all relations between them are illustrated.

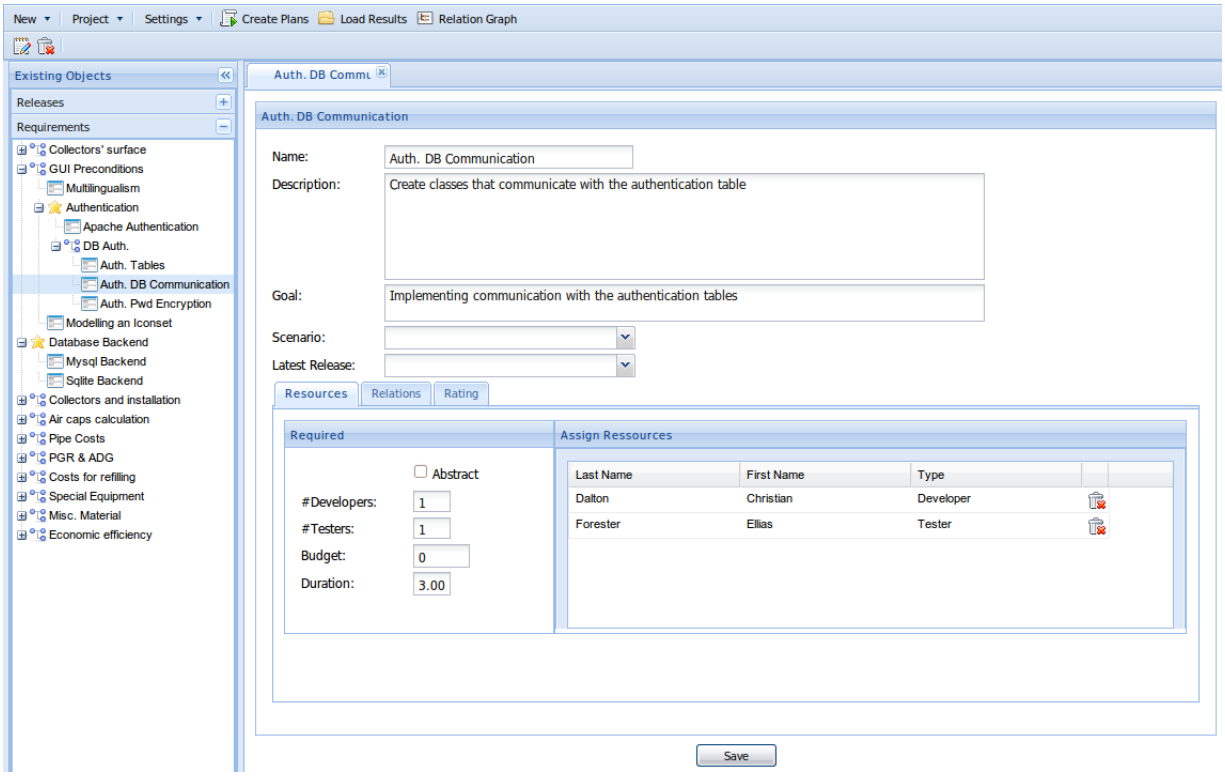


Figure 5.2: Defining a requirement with the prototype web interface.

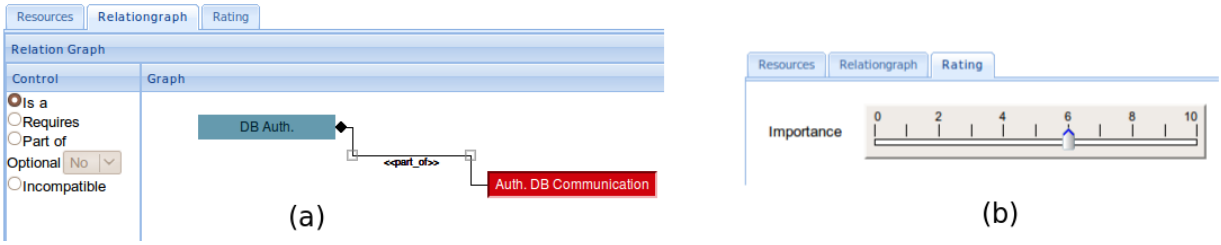


Figure 5.3: (a) Requirement relation editor and (b) requirement rating with the prototype web interface.

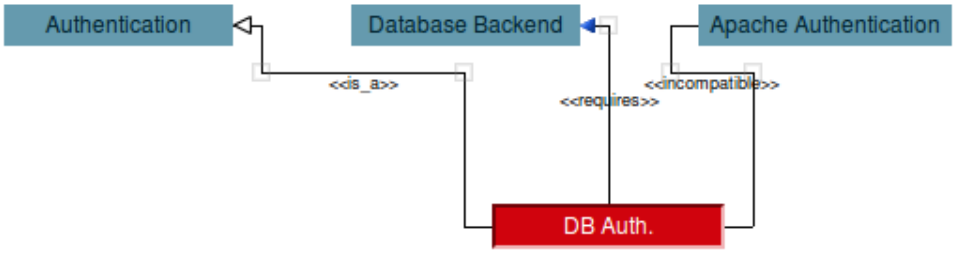


Figure 5.4: Multiple relations modelled with the requirement relation editor.

5.2 Release Plans

After the requirements and releases have been defined, the release plans can be generated. Figure 5.6 shows a form to parametrize the search. Since, constraint programming algorithms are complete all available solutions can be found. This can take a lot of time, and therefore it is not often desired. The prototype allows to set two types of limits. Either a maximum number of plans or a time limit can be defined. In Figure 5.6 a search shall stop after 20 plans were found or after 60 seconds at the latest. The last setting was already discussed in Section 4.4.3, it ensures that the sub-requirements of a decomposed requirement are scheduled close together. Release plans generated on the basis of the presented requirements model and the settings of Figure 5.6 are shown in Figure 5.7. It can be seen that 20 plans could be found. Measurements show that these 20 plans could be found within 7 seconds.

Id	Name	Latest Release	#Devel.	#Test.	Budget	Duration	Assigned Persons	Rating
0	Multilingualism	—	1	1	2000	9		5
1	Authentication	—	Abstract					5
2	Collectors' surface	—	Abstract					5
3	GUI Preconditions	—	Abstract					5
4	Collectors' surface (GUI)	—	1	1	500	2		5
5	Database Backend	—	Abstract					5
6	Collectors and installation	1	Abstract					5
7	Collectors's surface (Store)	—	1	1	100	3		5
8	Collectors and installation (Store)	—	1	1	0	2		5
9	Air caps calculation	—	Abstract					5
10	Air caps calculation (Store)	—	1	1	0	2		5
11	Pipe Costs	—	Abstract					5
12	Pipe Costs (Store)	—	1	1	0	2		5
13	PGR & ADG	2	Abstract					5
14	PGR & ADG (Store)	—	1	1	0	2		5
15	Costs for refilling	—	Abstract					5
16	Costs for refilling (Store)	—	1	1	0	2		5
17	Special Equipment	—	Abstract					5
18	Special Equipment (Store)	—	1	1	0	2		5
19	Misc. Material	—	Abstract					5
20	Misc. Material (Store)	—	2	1	0	3		5
21	Misc. Material (GUI)	—	2	2	500	3		5
22	Economic efficiency	3	Abstract					5
23	Collectors and installation (GUI)	—	1	1	500	2		5
24	Pipe costs (GUI-input)	—	1	1	500	2		5
25	Air caps calculation (GUI-Input)	—	1	1	500	2		5
26	PGR & ADG (GUI-Input)	—	2	1	500	2		5
27	Costs for refilling (Calculation)	—	3	2	0	3		5
28	Special Equipment (GUI-input)	—	1	1	500	3		5
29	PGR & ADG (Calculation)	—	3	2	0	10		5
30	Eco. efficiency (Calc.)	—	1	2	100	3		5
31	Eco. efficiency (GUI-input)	—	2	1	500	2		5
32	Eco. efficiency (Proposal)	—	2	1	100	14		5
33	Modelling an Iconset	—	3	2	2000	7		5
34	Mysql Backend	—	3	2	1500	5		10
35	Sqlite Backend	—	2	1	1000	2		1
36	Costs for refilling (GUI-input)	—	3	2	500	3		5
37	Apache Authentication	—	1	2	1000	3		1
38	DB Auth.	—	Abstract					10
39	Auth. Tables	—	1	1	0	2	Dalton Forester	10 10
40	Auth. DB Communication	—	1	1	0	3	Dalton Forester	10 10
41	Auth. Pwd Encryption	—	1	1	0	2	Dalton Forester	10 10
42	Air caps calculation (GUI-Results)	—	2	1	500	2		5
43	Pipe costs (GUI-results)	—	2	1	500	2		5
44	PGR & ADG (GUI-Result)	—	2	2	500	2		5
45	Special Equipment (GUI-results)	—	2	1	500	2		5
46	Special Equipment (Calculation)	—	1	2	0	2		5
47	Eco. efficiency (Graphics)	—	2	1	3600	14		10
48	Eco. efficiency (Proposal GUI)	—	2	1	500	2		5

Table 5.2: Basic data of the requirements of the example application.

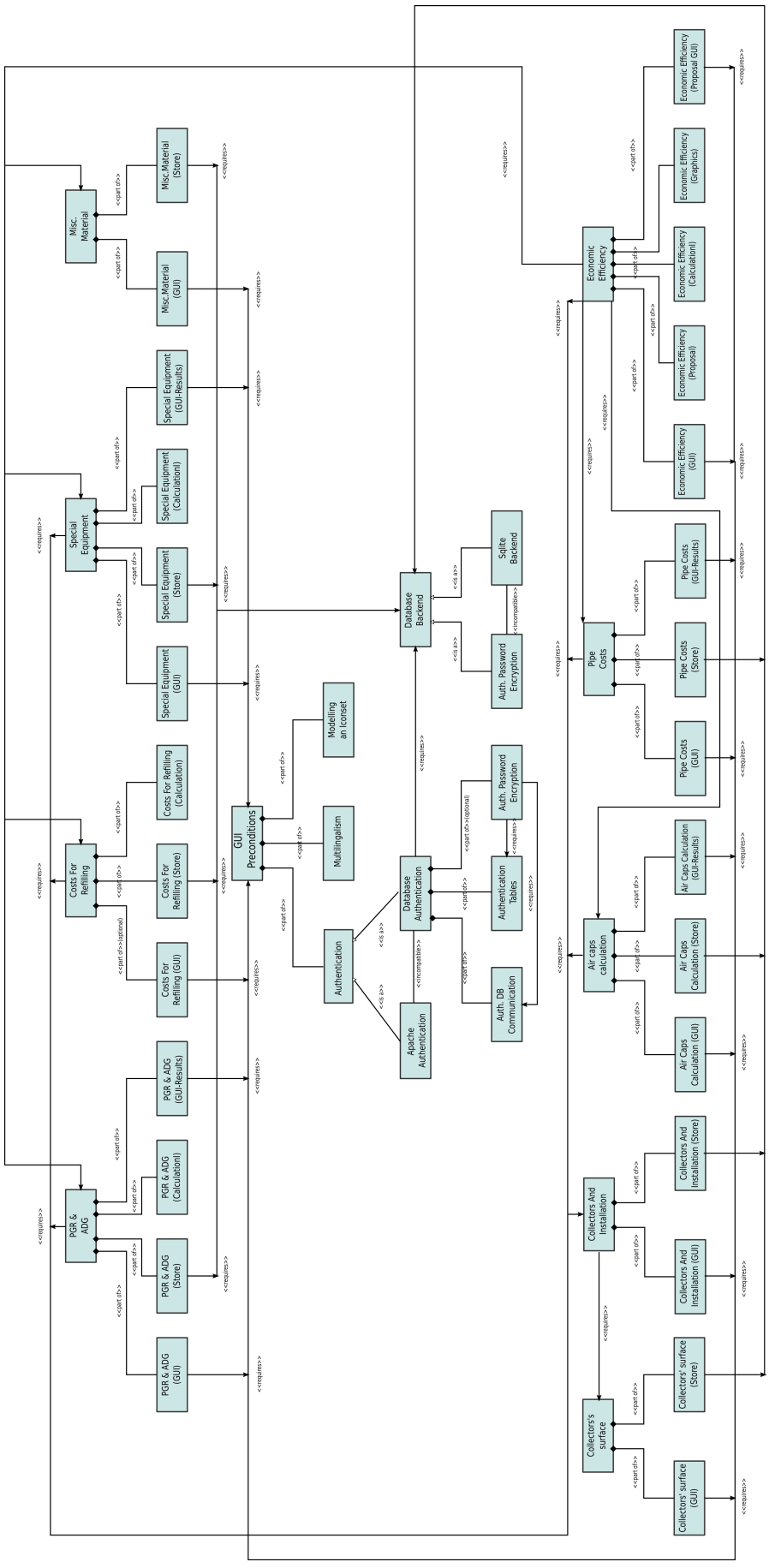


Figure 5.5: Requirement relations overview of the example application.

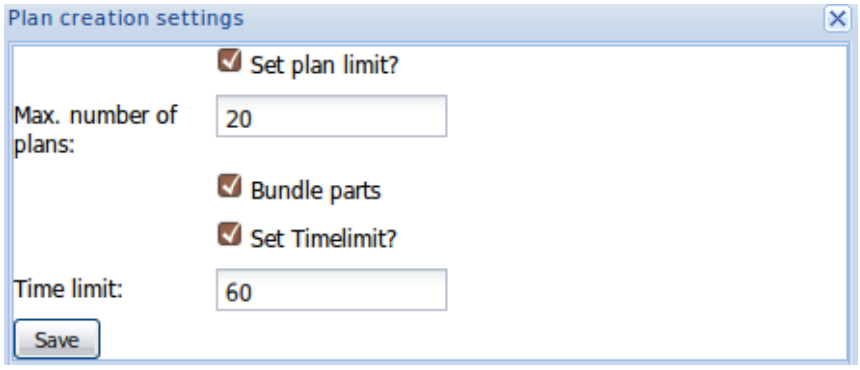


Figure 5.6: Search settings of the example application.

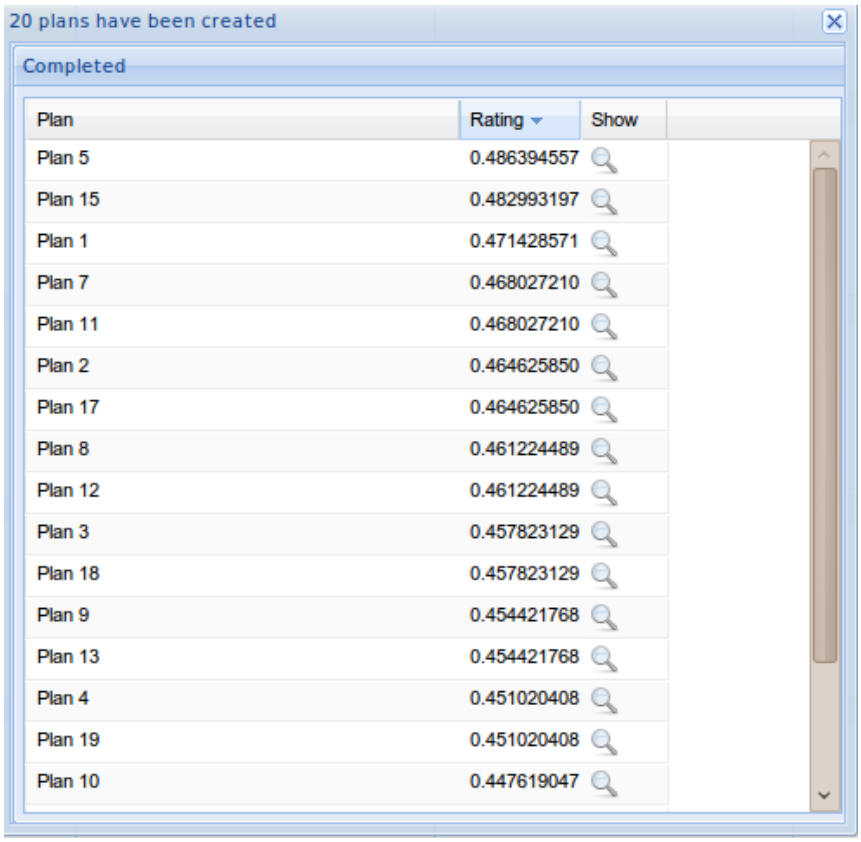


Figure 5.7: Generated release plans of the example application.

The plans of Figure 5.7 are named after their creation time. This means that the plan titled “Plan 1” was discovered as first plan. The plans are ordered by the ranking Formula 3. In the presented requirements model ‘Plan 5” got the best ranking, whereas “Plan 16” got the worst ranking. Later in this section the differences between these two plans are presented. The Figures 5.8 and 5.9 show the detailed representation of “Plan 5”. In

Figure 5.8 the requirements of the first release are illustrated. For a better overview, all requirements belonging to the first release are highlighted in blue. Figure 5.9 shows the requirements that shall be implemented in the second and third release. All requirements highlighted in red belong to the second release and all those highlighted in green belong to the third release. The requirements are labeled with their names and their resources are given in brackets. There, the first number indicates the required number of developers, the second one the number of testers, and the last number represents the required budget. Figure 5.10 shows a detailed view of the created "Auth. DB Communication" requirement. In addition to the description also the assigned resources are displayed. As expected this requirement has to be implemented by developer "Dalton" and tested by tester "Forester", see Table 5.2.

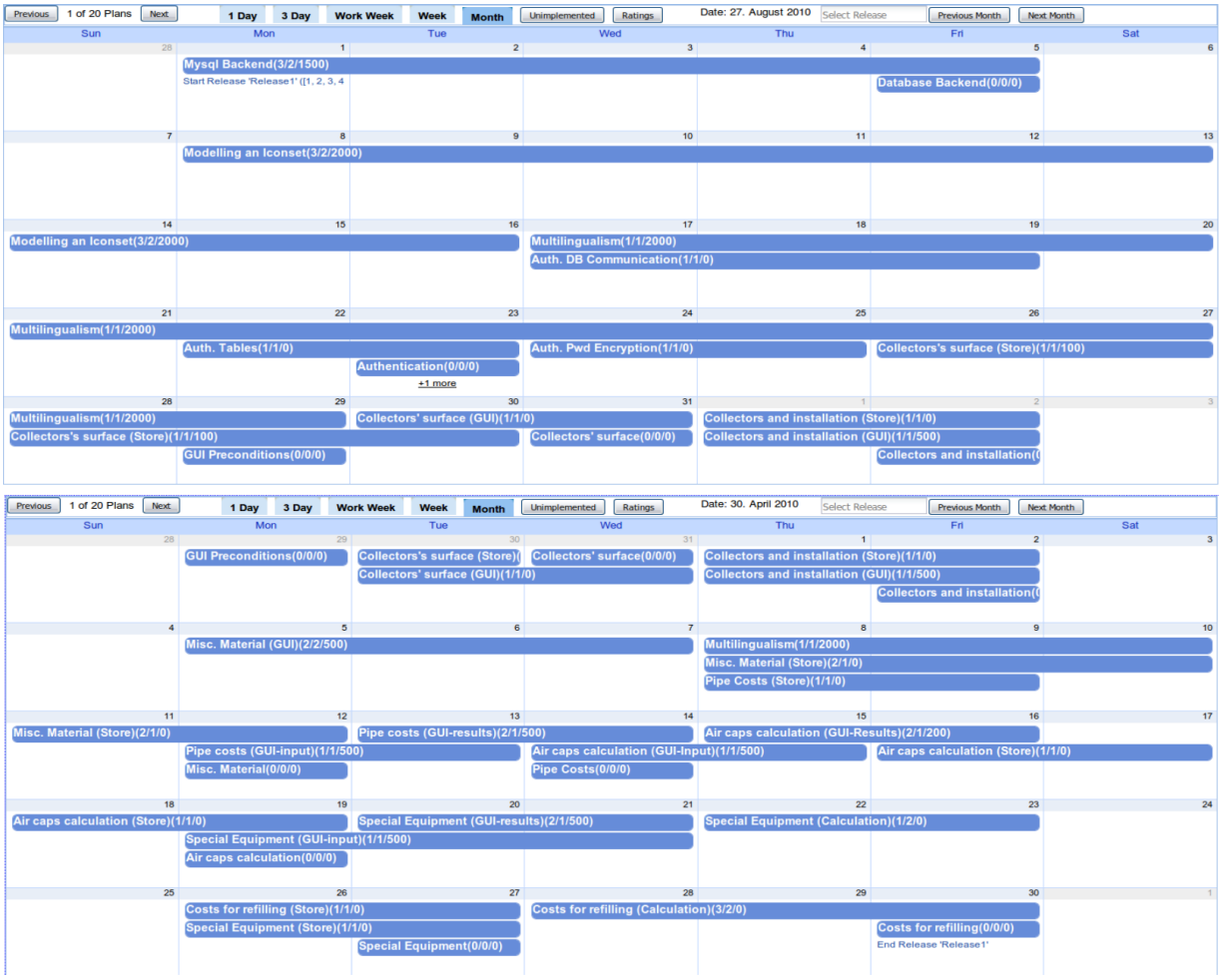


Figure 5.8: Prototype plan presentation: release 1

Figure 5.9: Prototype plan presentation: release 2 and release 3

Another feature implemented in the prototype is the possibility to compare plans. This shall illustrate that the proposed ranking reflects the quality of the plans. In Figure 5.11 the highest rated plan “Plan 5” is compared to the lowest rated one, “Plan 16”. Having a closer look to the requirements of Table 5.2 it can be seen that “Plan 5” implements

all high rated requirements like “DB Authentication” and “Mysql Backend”. “Plan 16” on the other hand schedules the low rated alternatives “Apache Authentication” and “Sqlite Backend”.

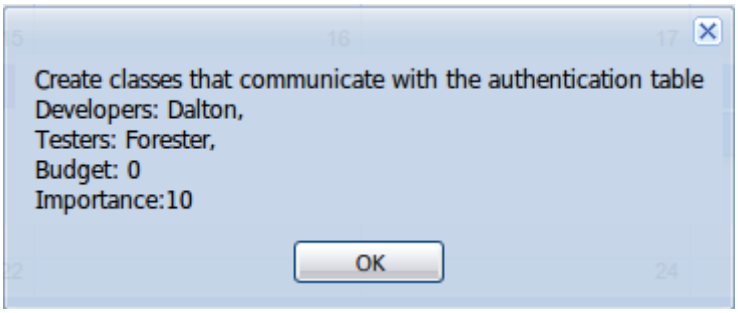
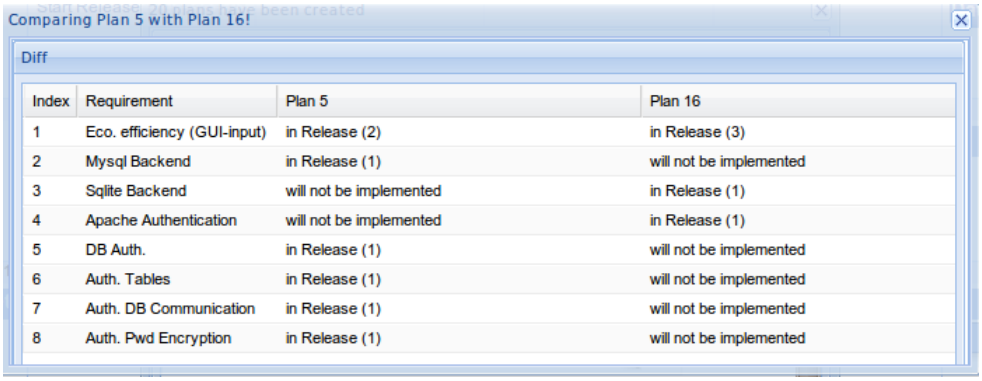


Figure 5.10: Detail view of requirement “Auth. DB Communication”



Index	Requirement	Plan 5	Plan 16
1	Eco. efficiency (GUI-input)	in Release (2)	in Release (3)
2	Mysql Backend	in Release (1)	will not be implemented
3	Sqlite Backend	will not be implemented	in Release (1)
4	Apache Authentication	will not be implemented	in Release (1)
5	DB Auth.	in Release (1)	will not be implemented
6	Auth. Tables	in Release (1)	will not be implemented
7	Auth. DB Communication	in Release (1)	will not be implemented
8	Auth. Pwd Encryption	in Release (1)	will not be implemented

Figure 5.11: Comparing highest rated plan to lowest rated one.

5.3 Diagnosis

As mentioned in Section 4.5 the prototype integrates a diagnosis framework to identify inconsistencies. There are different kinds of inconsistencies, like circular *require* relations or *incompatibilities* that do not lead to a solution. The prototype provides the possibility to start a diagnosis after a regular search has not found any solutions. How a diagnosis looks like shall be shown in an example. In Figure 5.5 the requirements “Collectors’ Surface (Store)” and “Collectors And Installation (Store)” requires a database backend. For demonstration purpose the requirements model is changed. Now “Collectors’ Surface (Store)” requires a Mysql backend and “Collectors And Installation (Store)” requires a

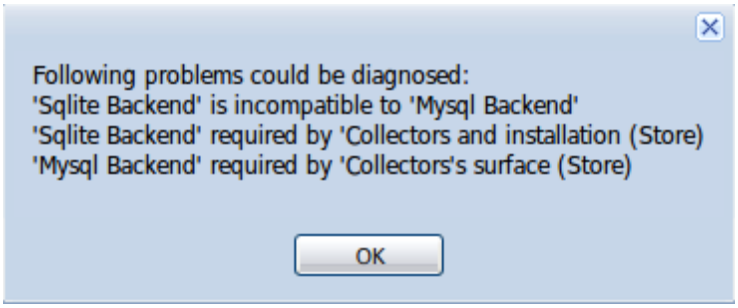


Figure 5.12: Diagnosing *incompatible* relations.

Sqlite backend. In the model these two are incompatible and therefore no solution can be found. Fortunately, such inconsistencies abort a search process immediately and a diagnosis can start. In Figure 5.12 the result of the diagnosis is illustrated. Inconsistencies can also occur because of circular *requires* relations. In the original requirements model, see Figure 5.5, the requirement "Air Caps Calculation" requires "Collectors And Installation". Now an additional *requires* relation is added, that express that "Collectors And Installation" requires "Air Caps Calculation". Because of this circular *requires* relation no solution can be found. The diagnosis of this inconsistency is shown in Figure 5.13. Before integrating the diagnosis framework, debugging the model was really hard, because the solver only provides the variables and the domains, but does not offer the constraints that lead to the inconsistencies.

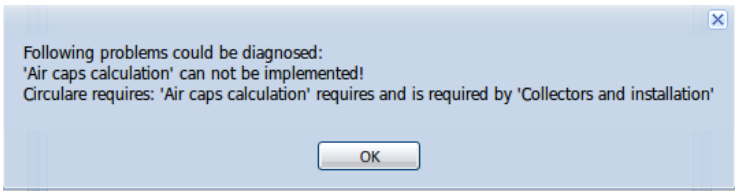


Figure 5.13: Diagnosing circular *require* relations.

Chapter 6

Conclusions

6.1 Results

In this master thesis the following objectives could be reached:

- A release planning task could be modelled as a constraint satisfaction problem. Therefore, variables and constraints of requirements engineering and release planning were identified and modelled with the help of the open source constraint programming library JaCoP[1].
- It could be shown that a requirements model, especially the relations and interdependencies between requirements can be modelled with standard configuration modelling languages.
- The successful integration of an existing model based diagnosis component allows to debug faulty requirements models by identifying inconsistencies.
- An introduced ranking algorithm allows to order the found release plans according to their overall stakeholders satisfaction.
- A web-based user interface was developed for creating requirements models and generating release plans. It includes an UML-oriented modelling environment to define relations between requirements. Furthermore, the integrated diagnosis component is used to display inconsistencies and recommends requirements that have to be changed in the requirements model.
- A reference requirements model consisting of almost 50 requirements and more than 80 relations was modelled by the proposed prototype. It could be shown that the generic algorithms of the constraint solver found 20 different plans in less than ten seconds. Furthermore, the found plans was ranked by the proposed ranking algorithm. Comparisons of the found plans showed that plans with a high stakeholder satisfaction could be found as well as low rated plans.

[1] http://jacop.osolpro.com/

- The implemented reference model also showed that the proposed release planner can be used with fixed start and end dates of a release as well as with flexible release ranges.

6.2 Future Work

Starting from the results achieved in this thesis several directions of future research is thinkable:

- Extending the proposed prototype with features that support groups during the whole requirements engineering process.
- Improvements of the presented ranking algorithm. Beside the proposed rating attribute *importance* other attributes may be useful to integrate into the ranking process.
- In addition to the proposed ranking algorithm, it is also thinkable to influence the search in a way that the high ranked release plans shall be found early.
- Integration of recommender algorithms in requirements engineering.
- Further performance evaluations by implementing a large requirements model.

List of Figures

List of Tables

Bibliography

Amandeep, N., Ruhe, G., and Stanford, M. (2003). Intelligent support for software release planning. *Product Focused Software Process Improvement*, pages 248–262.

Barták, R. (1999). Constraint programming: In pursuit of the holy grail. *Proceedings of the Week of Doctoral Students (WDS99), Part IV, MatFyzPress*, pages 555–564.

Birrell, A. and Nelson, B. (1984). Implementing remote procedure calls. *ACM Transactions on Computer Systems (TOCS)*, 2(1):39–59.

Boehm, B., Bose, P., Horowitz, E., and Lee, M. (1995). Software requirements negotiation and renegotiation aids: A theory-W based spiral approach. In *Proc. ICSE-17 - 17th Intl. Conf. on Software Engineering*, pages 243–253.

Borning, A. (1981). The programming language aspects of ThingLab, a constraint-oriented simulation laboratory. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 3(4):387.

Bosch, J. (2000). *Design and use of software architectures: adopting and evolving a product-line approach*. Addison-Wesley Professional.

Cooper, M. and Schiex, T. (2004). Arc consistency for soft constraints. *Artificial Intelligence*, 154(1-2):199–227.

Felfernig, A., Friedrich, G., and Jannach, D. (2000). UML as domain specific language for the construction of knowledge-based configuration systems. *International Journal of Software Engineering and Knowledge Engineering*, 10(4):449–470.

Felfernig, A., Friedrich, G., Jannach, D., and Stumptner, M. (2004). Consistency-based diagnosis of configuration knowledge bases. *Artificial Intelligence*, 152(2):213–234.

Felfernig, A., Friedrich, G., Jannach, D., and Zanker, M. (2006). An integrated environment for the development of knowledge-based recommender applications. *International Journal of Electronic Commerce*, 11(2):11–34.

Felfernig, A., Friedrich, G., Schubert, M., Mandl, M., Mairitsch, M., and Teppan, E. (2009). Plausible repairs for inconsistent requirements. In *21st International Joint Conference on Artificial Intelligence (IJCAI'09), Pasadena, California, USA*, pages 791–796.

Fikes, R. (1970). REF-ARF: A system for solving problems stated as procedures. *Artificial Intelligence*, 1(1-2):27–120.

Gallaire, H. (1985). Logic Programming: Further Developments. In *IEEE Symposium on Logic Programming*, pages 88–99.

Garrett, J. et al. (2005). Ajax: A new approach to web applications. *http://www.adaptivepath.com/ideas/essays/archives/000385.php*.

Goknil, A., Kurtev, I., and van den Berg, K. (2010). A metamodeling approach for reasoning about requirements.

Gotel, O. and Finkelstein, A. (1994). An analysis of the requirements traceability problem. In *Proceedings of the First International Conference on Requirements Engineering*, pages 94–101.

Greer, D. and Ruhe, G. (2004). Software release planning: an evolutionary and iterative approach. *Information and Software Technology*, 46(4):243–253.

J. Jaffar, J. L. (1987). Constraint Logic Programming. In *Proc. The ACM Symposium on Principles of Programming Languages*, pages 111–119.

Junker, U. (2004). Preferred Explanations and Relaxations for Over-Constrained Problems. In *AAAI'04, San Jose, AAAI Press*, pages 167–172.

Kuchcinski, K. and Wolinski, C. (2003). Global approach to assignment and scheduling of complex behaviors based on HCDG and constraint programming. *Journal of systems architecture*, 49(12-15):489–503.

Lauriere, J. (1978). A language and a program for stating and solving combinatorial problems. *Artificial Intelligence*, 10(1):29–127.

Lin, J., Fox, M., and Bilgic, T. (1996). A requirement ontology for engineering design. *Concurrent Engineering: Research and Applications*, 4(3):279.

Lohmann, S., Riechert, T., Auer, S., and Ziegler, J. (2008). Collaborative development of knowledge bases in distributed requirements elicitation. *Software Engineering 2008 - Workshopband*, pages 22–28.

Maiden, N. (1998). CREWS-SAVRE: Scenarios for acquiring and validating requirements. *Automated Software Engineering*, 5(4):419–446.

Marriott, K. and Stuckey, P. (1998). *Programming with constraints: an introduction*. MIT press.

Ncube, C. and Maiden, N. (1999). PORE: Procurement-oriented requirements engineering method for the component-based systems engineering development paradigm. In *International Workshop on Component-Based Software Engineering, Los Angeles, California,USA*, pages 1–12.

Nuseibeh, B. and Easterbrook, S. (2000). Requirements engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 35–46.

Object Management Group (2006). OMG SysML Specification. *http://www.sysml.org/specs.htm*.

Popper, K. (2002). *Conjectures and refutations: The growth of scientific knowledge*. Routledge.

Pretschner, A., Pretschner, E., and Loetzbeyer, H. (2001). Model based testing with constraint logic programming: First results and challenges. In *In 2nd ICSE Intl. Workshop on Automated Program Analysis, Testing, and Veri (WAPATV'01)*.

Reiter, R. (1987). A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95.

Riechert, T., Lauenroth, K., and Lehmann, J. (2007). Semantisch unterstuetztes Requirements Engineering. In *Conference on Social Semantic Web(CSSW '07), Leipzig, Germany, 2007*.

Rossi, F., Van Beek, P., and Walsh, T. (2006a). Handbook of constraint programming. pages 85–118. Elsevier Science Ltd.

Rossi, F., Van Beek, P., and Walsh, T. (2006b). Handbook of constraint programming. pages 168–208. Elsevier Science Ltd.

Ruhe, G. (2005). Handbook of software engineering and knowledge engineering. pages 365–395. World Scientific Pub. Co.

Ruhe, G. and Saliu, M. (2005). The art and science of software release planning. *IEEE software*, 22(6):47–53.

Russell, S., Norvig, P., Canny, J., Malik, J., and Edwards, D. (1995). *Artificial Intelligence: a modern approach*. Prentice hall Englewood Cliffs, NJ.

Saraswat, V., Rinard, M., and Panangaden, P. (1991). The semantic foundations of concurrent constraint programming. In *Proceedings of the 18th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 333–352.

Sommerville, I. (1996). *Software Engineering. 5th Edition*.

Sommerville, I. and Kotonya, G. (1998). *Requirements engineering: processes and techniques*. John Wiley & Sons, Inc. New York, NY, USA.

Sussman, G., Steele, G., et al. (1980). Constraints–A language for expressing almost-hierarchical descriptions. *Artificial intelligence*, 14(1):1–39.

Sutherland, I. (1963). Sketchpad a man-machine graphical communication system. In *Proc. of AFIPS Spring Joint Comp. Conf*, volume 23, pages 329–346.

Van Beek, P. and Chen, X. (1999). CPlan: A constraint programming approach to planning. In *Proceedings of the national conference on artificial intelligence*, pages 585–590.

van Lamsweerde, A. (2004). Goal-oriented requirements engineering: A roundtrip from research to practice. In *Proceedings of the Requirements Engineering Conference, 12th IEEE International (RE'04)*, pages 4–7.

Van Lamsweerde, A., Darimont, R., and Letier, E. (1998). Managing conflicts in goal-driven requirements engineering. *IEEE Transactions on Software Engineering*, 24(11):908–926.

Vicente-Chicote, C., Moros, B., and Toval, A. (2007). Remm-studio: an integrated model-driven environment for requirements specification, validation and formatting. *Journal of Object Technology*, 6(9):437–454.

Wallace, M. (1996). Practical applications of constraint programming. *Constraints*, 1(1):139–168.

White, J. (1976). A high-level framework for network-based resource sharing. In *Proceedings of the national computer conference and exposition, 1976*, pages 561–570.

Wieringa, R. (1996). *Requirements engineering: frameworks for understanding*. John Wiley & Sons.

Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Computing Surveys (CSUR)*, 29(4):315–321.

Nomenclature

AC	Arc Consistency
AJAX	Asynchronous JavaScript and XML
BC	Backchecking
BFS	Breadth-First Search
BJ	Backjumping
BM	Backmarking
CGI	Common Gateway Interfaces
CP	Constraint Programming
CS	Conflict Set
CSP	Constraint Satisfaction Problems
ER-Diagrams	Entity-relation diagrams
FC	Forward Checking
FDV	Finite Domain Variable
GUI	Graphical User Interface
GWT	Google Web Toolkit
HSDAG	Hitting Set Directed Acyclic Graph
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
JaCoP	Java Constraint Programming

JDBC	Java Database Connectivity
MAC	Maintaining Arc Consistency
MAUT	Multi Attribute Decision Theory
MRV	Minimum Remaining Values
NC	Node Consistency
RE	Requirements Engineering
RP	Release Planning
RPC	Remote Procedure Call
RT	requirement traceability
SWORE	SoftWiki Ontology for RE
UML	Unified modelling Language
XP	Extreme Programming

Index

Appendix A

FDVs Of The Example Application

Id	Name	Start	Duration	End
Rel0	Release1	0	45	45
Rel1	Release2	45	21..25	66..70
Rel2	Release3	66..70	25..34	95..100

Table A.1: Release FDVs for the example application

Id	Name	Start	Duration	End	Bounded developers	Needed developers	Used developers	Bounded testers	Needed testers	Used testers	Bounded budget	Needed budget	Used budget	Assigned release	Release duration of requirement	Release start of requirement
Req0	Multilingualism	0..223	9	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	2000	2000..17600	0..3	0..232	0..100
Req1	Authentication	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req2	Collectors' surface	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req3	GUI Preconditions	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req4	Collectors' surface (GUI)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req5	Database Backend	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req6	Collectors and installation	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req7	Collectors's surface (Store)	0..229	3	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	100	100..17600	0..3	0..232	0..100
Req8	Collectors and installation (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req9	Air caps calculation	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req10	Air caps calculation (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req11	Pipe Costs	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req12	Pipe Costs (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req13	PGR & ADG	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req14	PGR & ADG (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req15	Costs for refilling	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req16	Costs for refilling (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req17	Special Equipment	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req18	Special Equipment (Store)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req19	Misc. Material	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req20	Misc. Material (Store)	0..229	3	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req21	Misc. Material (GUI)	0..229	3	0..232	0..3	2	2..3	0..2	2	2	0..17600	500	500..17600	0..3	0..232	0..100
Req22	Economic efficiency	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req23	Collectors and installation (GUI)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req24	Pipe costs (GUI-input)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req25	Air caps calculation (GUI-Input)	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req26	PGR & ADG (GUI-Input)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req27	Costs for refilling (Calculation)	0..229	3	0..232	0..3	3	3	0..2	2	2	0..17600	0	0..17600	0..3	0..232	0..100
Req28	Special Equipment (GUI-input)	0..229	3	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req29	PGR & ADG (Calculation)	0..222	10	0..232	0..3	3	3	0..2	2	2	0..17600	0	0..17600	0..3	0..232	0..100
Req30	Eco. efficiency (Calc.)	0..229	3	0..232	0..3	1	1..3	0..2	2	2	0..17600	100	100..17600	0..3	0..232	0..100
Req31	Eco. efficiency (GUI-input)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req32	Eco. efficiency (Proposal)	0..218	14	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	100	100..17600	0..3	0..232	0..100
Req33	Modelling an Iconset	0..225	7	0..232	0..3	3	3	0..2	2	2	0..17600	2000	2000..17600	0..3	0..232	0..100
Req34	Mysql Backend	0..227	5	0..232	0..3	3	3	0..2	2	2	0..17600	1500	1500..17600	0..3	0..232	0..100
Req35	Sqlite Backend	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	1000	1000..17600	0..3	0..232	0..100
Req36	Costs for refilling (GUI-input)	0..229	3	0..232	0..3	3	3	0..2	2	2	0..17600	500	500..17600	0..3	0..232	0..100
Req37	Apache Authentication	0..229	3	0..232	0..3	1	1..3	0..2	2	2	0..17600	1000	1000..17600	0..3	0..232	0..100
Req38	DB Auth.	0..232	0	0..232	0..3	0	0..3	0..2	0	0..2	0..17600	0	0..17600	0..3	0..232	0..100
Req39	Auth. Tables	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req40	Auth. DB Communication	0..229	3	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req41	Auth. Pwd Encryption	0..230	2	0..232	0..3	1	1..3	0..2	1	1..2	0..17600	0	0..17600	0..3	0..232	0..100
Req42	Air caps calculation (GUI-Results)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	200	200..17600	0..3	0..232	0..100
Req43	Pipe costs (GUI-results)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req44	PGR & ADG (GUI-Result)	0..230	2	0..232	0..3	2	2..3	0..2	2	2	0..17600	500	500..17600	0..3	0..232	0..100
Req45	Special Equipment (GUI-results)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100
Req46	Special Equipment (Calculation)	0..230	2	0..232	0..3	1	1..3	0..2	2	2	0..17600	0	0..17600	0..3	0..232	0..100
Req47	Eco. efficiency (Graphics)	0..218	14	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	3600	3600..17600	0..3	0..232	0..100
Req48	Eco. efficiency (Proposal GUI)	0..230	2	0..232	0..3	2	2..3	0..2	1	1..2	0..17600	500	500..17600	0..3	0..232	0..100

Table A.2: Requirement implementation FDVs for the example application.

Appendix B

Constraints Of The Example Application

Constraints for the start of the project and overlapping releases.

Project End:	Max(project_end, {Rel0→end, Rel1→end, Rel2→end})
At least one requirement starts on day 0:	Min(0, {Req0→start, Req1→start, Req2→start, Req3→start, Req4→start, Req5→start, Req6→start, Req7→start, Req8→start, Req9→start, Req10→start, Req11→start, Req12→start, Req13→start, Req14→start, Req15→start, Req16→start, Req17→start, Req18→start, Req19→start, Req20→start, Req21→start, Req22→start, Req23→start, Req24→start, Req25→start, Req26→start, Req27→start, Req28→start, Req29→start, Req30→start, Req31→start, Req32→start, Req33→start, Req34→start, Req35→start, Req36→start, Req37→start, Req38→start, Req39→start, Req40→start, Req41→start, Req42→start, Req43→start, Req44→start, Req45→start, Req46→start, Req47→start, Req48→start})
At least one release starts on day 0:	Min(0, {Rel0→start, Rel1→start, Rel2→start})
Release2 starts after Release1 ends:	XgteqY(Rel1→start, Rel0→end)
Release3 starts after Release2 ends:	XgteqY(Rel2→start, Rel1→end)

Table B.1: Example application: Determining the project end, ensuring that one release and at least one requirement implementation starts on the first day, and releases must not overlap.

Constraints for start, duration, and end time of releases and implementations

Multilingualism:	XplusYeqZ(Req0→start, Req0→duration, Req0→end)
Authentication:	XplusYeqZ(Req1→start, Req1→duration, Req1→end)
Collectors' surface:	XplusYeqZ(Req2→start, Req2→duration, Req2→end)
GUI Preconditions:	XplusYeqZ(Req3→start, Req3→duration, Req3→end)
Collectors' surface (GUI):	XplusYeqZ(Req4→start, Req4→duration, Req4→end)
Database Backend:	XplusYeqZ(Req5→start, Req5→duration, Req5→end)
Collectors and installation:	XplusYeqZ(Req6→start, Req6→duration, Req6→end)
Collectors's surface (Store):	XplusYeqZ(Req7→start, Req7→duration, Req7→end)
Collectors and installation (Store):	XplusYeqZ(Req8→start, Req8→duration, Req8→end)
Air caps calculation:	XplusYeqZ(Req9→start, Req9→duration, Req9→end)
Air caps calculation (Store):	XplusYeqZ(Req10→start, Req10→duration, Req10→end)
Pipe Costs:	XplusYeqZ(Req11→start, Req11→duration, Req11→end)
Pipe Costs (Store):	XplusYeqZ(Req12→start, Req12→duration, Req12→end)
PGR & ADG:	XplusYeqZ(Req13→start, Req13→duration, Req13→end)
PGR & ADG (Store):	XplusYeqZ(Req14→start, Req14→duration, Req14→end)
Costs for refilling:	XplusYeqZ(Req15→start, Req15→duration, Req15→end)
Costs for refilling (Store):	XplusYeqZ(Req16→start, Req16→duration, Req16→end)
Special Equipment:	XplusYeqZ(Req17→start, Req17→duration, Req17→end)
Special Equipment (Store):	XplusYeqZ(Req18→start, Req18→duration, Req18→end)
Misc. Material:	XplusYeqZ(Req19→start, Req19→duration, Req19→end)
Misc. Material (Store):	XplusYeqZ(Req20→start, Req20→duration, Req20→end)
Misc. Material (GUI):	XplusYeqZ(Req21→start, Req21→duration, Req21→end)
Economic efficiency:	XplusYeqZ(Req22→start, Req22→duration, Req22→end)
Collectors and installation (GUI):	XplusYeqZ(Req23→start, Req23→duration, Req23→end)
Pipe costs (GUI-input):	XplusYeqZ(Req24→start, Req24→duration, Req24→end)
Air caps calculation (GUI-Input):	XplusYeqZ(Req25→start, Req25→duration, Req25→end)
PGR & ADG (GUI-Input):	XplusYeqZ(Req26→start, Req26→duration, Req26→end)
Costs for refilling (Calculation):	XplusYeqZ(Req27→start, Req27→duration, Req27→end)
Special Equipment (GUI-input):	XplusYeqZ(Req28→start, Req28→duration, Req28→end)
PGR & ADG (Calculation) :	XplusYeqZ(Req29→start, Req29→duration, Req29→end)
Eco. efficiency (Calc.):	XplusYeqZ(Req30→start, Req30→duration, Req30→end)
Eco. efficiency (GUI-input):	XplusYeqZ(Req31→start, Req31→duration, Req31→end)
Eco. efficiency (Proposal):	XplusYeqZ(Req32→start, Req32→duration, Req32→end)
Modelling an Iconset:	XplusYeqZ(Req33→start, Req33→duration, Req33→end)
Mysql Backend:	XplusYeqZ(Req34→start, Req34→duration, Req34→end)
Sqlite Backend:	XplusYeqZ(Req35→start, Req35→duration, Req35→end)
Costs for refilling (GUI-input):	XplusYeqZ(Req36→start, Req36→duration, Req36→end)
Apache Authentication:	XplusYeqZ(Req37→start, Req37→duration, Req37→end)
DB Auth.:	XplusYeqZ(Req38→start, Req38→duration, Req38→end)
Auth. Tables:	XplusYeqZ(Req39→start, Req39→duration, Req39→end)
Auth. DB Communication:	XplusYeqZ(Req40→start, Req40→duration, Req40→end)
Auth. Pwd Encryption:	XplusYeqZ(Req41→start, Req41→duration, Req41→end)
Air caps calculation (GUI-Results):	XplusYeqZ(Req42→start, Req42→duration, Req42→end)
Pipe costs (GUI-results):	XplusYeqZ(Req43→start, Req43→duration, Req43→end)
PGR & ADG (GUI-Result):	XplusYeqZ(Req44→start, Req44→duration, Req44→end)
Special Equipment (GUI-results):	XplusYeqZ(Req45→start, Req45→duration, Req45→end)
Special Equipment (Calculation):	XplusYeqZ(Req46→start, Req46→duration, Req46→end)
Eco. efficiency (Graphics):	XplusYeqZ(Req47→start, Req47→duration, Req47→end)
Eco. efficiency (Proposal GUI):	XplusYeqZ(Req48→start, Req48→duration, Req48→end)
Release1:	XplusYeqZ(Rel0→start, Rel0→duration, Rel0→end)
Release2:	XplusYeqZ(Rel1→start, Rel1→duration, Rel1→end)
Release3:	XplusYeqZ(Rel2→start, Rel2→duration, Rel2→end)

Table B.2: Example Application: Relations between start, duration, and end of releases and requirement implementations.

Mandatory requirements

'Collectors' surface' is mandatory	XltC(Req2→assigned_release,3)
'GUI Preconditions' is mandatory	XltC(Req3→assigned_release,3)
'Database Backend' is mandatory	XltC(Req5→assigned_release,3)
'Collectors and installation' is mandatory	XltC(Req6→assigned_release,3)
'Air caps calculation' is mandatory	XltC(Req9→assigned_release,3)
'Pipe Costs' is mandatory	XltC(Req11→assigned_release,3)
'PGR & ADG' is mandatory	XltC(Req13→assigned_release,3)
'Costs for refilling' is mandatory	XltC(Req15→assigned_release,3)
'Special Equipment' is mandatory	XltC(Req17→assigned_release,3)
'Misc. Material' is mandatory	XltC(Req19→assigned_release,3)
'Economic efficiency' is mandatory	XltC(Req22→assigned_release,3)

Table B.3: Example Application: Mandatory requirements.

Constraints for release starts and ends

Possible starts of Release1 : Or
(XeqY(Rel0→start,Req0→start), XeqY(Rel0→start,Req1→start), XeqY(Rel0→start,Req2→start),
XeqY(Rel0→start,Req3→start), XeqY(Rel0→start,Req4→start), XeqY(Rel0→start,Req5→start),
XeqY(Rel0→start,Req6→start), XeqY(Rel0→start,Req7→start), XeqY(Rel0→start,Req8→start),
XeqY(Rel0→start,Req9→start), XeqY(Rel0→start,Req10→start), XeqY(Rel0→start,Req11→start),
XeqY(Rel0→start,Req12→start), XeqY(Rel0→start,Req13→start), XeqY(Rel0→start,Req14→start),
XeqY(Rel0→start,Req15→start), XeqY(Rel0→start,Req16→start), XeqY(Rel0→start,Req17→start),
XeqY(Rel0→start,Req18→start), XeqY(Rel0→start,Req19→start), XeqY(Rel0→start,Req20→start),
XeqY(Rel0→start,Req21→start), XeqY(Rel0→start,Req22→start), XeqY(Rel0→start,Req23→start),
XeqY(Rel0→start,Req24→start), XeqY(Rel0→start,Req25→start), XeqY(Rel0→start,Req26→start),
XeqY(Rel0→start,Req27→start), XeqY(Rel0→start,Req28→start), XeqY(Rel0→start,Req29→start),
XeqY(Rel0→start,Req30→start), XeqY(Rel0→start,Req31→start), XeqY(Rel0→start,Req32→start),
XeqY(Rel0→start,Req33→start), XeqY(Rel0→start,Req34→start), XeqY(Rel0→start,Req35→start),
XeqY(Rel0→start,Req36→start), XeqY(Rel0→start,Req37→start), XeqY(Rel0→start,Req38→start),
XeqY(Rel0→start,Req39→start), XeqY(Rel0→start,Req40→start), XeqY(Rel0→start,Req41→start),
XeqY(Rel0→start,Req42→start), XeqY(Rel0→start,Req43→start), XeqY(Rel0→start,Req44→start),
XeqY(Rel0→start,Req45→start), XeqY(Rel0→start,Req46→start), XeqY(Rel0→start,Req47→start),
XeqY(Rel0→start,Req48→start),)

Possible starts of Release2 : Or
(XeqY(Rel1→start,Req0→start), XeqY(Rel1→start,Req1→start), XeqY(Rel1→start,Req2→start),
XeqY(Rel1→start,Req3→start), XeqY(Rel1→start,Req4→start), XeqY(Rel1→start,Req5→start),
XeqY(Rel1→start,Req6→start), XeqY(Rel1→start,Req7→start), XeqY(Rel1→start,Req8→start),
XeqY(Rel1→start,Req9→start), XeqY(Rel1→start,Req10→start), XeqY(Rel1→start,Req11→start),
XeqY(Rel1→start,Req12→start), XeqY(Rel1→start,Req13→start), XeqY(Rel1→start,Req14→start),
XeqY(Rel1→start,Req15→start), XeqY(Rel1→start,Req16→start), XeqY(Rel1→start,Req17→start),
XeqY(Rel1→start,Req18→start), XeqY(Rel1→start,Req19→start), XeqY(Rel1→start,Req20→start),
XeqY(Rel1→start,Req21→start), XeqY(Rel1→start,Req22→start), XeqY(Rel1→start,Req23→start),
XeqY(Rel1→start,Req24→start), XeqY(Rel1→start,Req25→start), XeqY(Rel1→start,Req26→start),
XeqY(Rel1→start,Req27→start), XeqY(Rel1→start,Req28→start), XeqY(Rel1→start,Req29→start),
XeqY(Rel1→start,Req30→start), XeqY(Rel1→start,Req31→start), XeqY(Rel1→start,Req32→start),
XeqY(Rel1→start,Req33→start), XeqY(Rel1→start,Req34→start), XeqY(Rel1→start,Req35→start),
XeqY(Rel1→start,Req36→start), XeqY(Rel1→start,Req37→start), XeqY(Rel1→start,Req38→start),
XeqY(Rel1→start,Req39→start), XeqY(Rel1→start,Req40→start), XeqY(Rel1→start,Req41→start),
XeqY(Rel1→start,Req42→start), XeqY(Rel1→start,Req43→start), XeqY(Rel1→start,Req44→start),
XeqY(Rel1→start,Req45→start), XeqY(Rel1→start,Req46→start), XeqY(Rel1→start,Req47→start),
XeqY(Rel1→start,Req48→start),)

Possible starts of Release3 : Or
(XeqY(Rel2→start,Req0→start), XeqY(Rel2→start,Req1→start), XeqY(Rel2→start,Req2→start),
XeqY(Rel2→start,Req3→start), XeqY(Rel2→start,Req4→start), XeqY(Rel2→start,Req5→start),
XeqY(Rel2→start,Req6→start), XeqY(Rel2→start,Req7→start), XeqY(Rel2→start,Req8→start),
XeqY(Rel2→start,Req9→start), XeqY(Rel2→start,Req10→start), XeqY(Rel2→start,Req11→start),
XeqY(Rel2→start,Req12→start), XeqY(Rel2→start,Req13→start), XeqY(Rel2→start,Req14→start),
XeqY(Rel2→start,Req15→start), XeqY(Rel2→start,Req16→start), XeqY(Rel2→start,Req17→start),
XeqY(Rel2→start,Req18→start), XeqY(Rel2→start,Req19→start), XeqY(Rel2→start,Req20→start),
XeqY(Rel2→start,Req21→start), XeqY(Rel2→start,Req22→start), XeqY(Rel2→start,Req23→start),
XeqY(Rel2→start,Req24→start), XeqY(Rel2→start,Req25→start), XeqY(Rel2→start,Req26→start),
XeqY(Rel2→start,Req27→start), XeqY(Rel2→start,Req28→start), XeqY(Rel2→start,Req29→start),
XeqY(Rel2→start,Req30→start), XeqY(Rel2→start,Req31→start), XeqY(Rel2→start,Req32→start),
XeqY(Rel2→start,Req33→start), XeqY(Rel2→start,Req34→start), XeqY(Rel2→start,Req35→start),
XeqY(Rel2→start,Req36→start), XeqY(Rel2→start,Req37→start), XeqY(Rel2→start,Req38→start),
XeqY(Rel2→start,Req39→start), XeqY(Rel2→start,Req40→start), XeqY(Rel2→start,Req41→start),
XeqY(Rel2→start,Req42→start), XeqY(Rel2→start,Req43→start), XeqY(Rel2→start,Req44→start),
XeqY(Rel2→start,Req45→start), XeqY(Rel2→start,Req46→start), XeqY(Rel2→start,Req47→start),
XeqY(Rel2→start,Req48→start),)

Possible ends of Release1	:	Or (XeqY(Rel0→end,Req0→end), XeqY(Rel0→end,Req1→end), XeqY(Rel0→end,Req2→end), XeqY(Rel0→end,Req3→end), XeqY(Rel0→end,Req4→end), XeqY(Rel0→end,Req5→end), XeqY(Rel0→end,Req6→end), XeqY(Rel0→end,Req7→end), XeqY(Rel0→end,Req8→end), XeqY(Rel0→end,Req9→end), XeqY(Rel0→end,Req10→end), XeqY(Rel0→end,Req11→end), XeqY(Rel0→end,Req12→end), XeqY(Rel0→end,Req13→end), XeqY(Rel0→end,Req14→end), XeqY(Rel0→end,Req15→end), XeqY(Rel0→end,Req16→end), XeqY(Rel0→end,Req17→end), XeqY(Rel0→end,Req18→end), XeqY(Rel0→end,Req19→end), XeqY(Rel0→end,Req20→end), XeqY(Rel0→end,Req21→end), XeqY(Rel0→end,Req22→end), XeqY(Rel0→end,Req23→end), XeqY(Rel0→end,Req24→end), XeqY(Rel0→end,Req25→end), XeqY(Rel0→end,Req26→end), XeqY(Rel0→end,Req27→end), XeqY(Rel0→end,Req28→end), XeqY(Rel0→end,Req29→end), XeqY(Rel0→end,Req30→end), XeqY(Rel0→end,Req31→end), XeqY(Rel0→end,Req32→end), XeqY(Rel0→end,Req33→end), XeqY(Rel0→end,Req34→end), XeqY(Rel0→end,Req35→end), XeqY(Rel0→end,Req36→end), XeqY(Rel0→end,Req37→end), XeqY(Rel0→end,Req38→end), XeqY(Rel0→end,Req39→end), XeqY(Rel0→end,Req40→end), XeqY(Rel0→end,Req41→end), XeqY(Rel0→end,Req42→end), XeqY(Rel0→end,Req43→end), XeqY(Rel0→end,Req44→end), XeqY(Rel0→end,Req45→end), XeqY(Rel0→end,Req46→end), XeqY(Rel0→end,Req47→end), XeqY(Rel0→end,Req48→end), XeqC(Rel 0→end,45))
Possible ends of Release2	:	Or (XeqY(Rel1→end,Req0→end), XeqY(Rel1→end,Req1→end), XeqY(Rel1→end,Req2→end), XeqY(Rel1→end,Req3→end), XeqY(Rel1→end,Req4→end), XeqY(Rel1→end,Req5→end), XeqY(Rel1→end,Req6→end), XeqY(Rel1→end,Req7→end), XeqY(Rel1→end,Req8→end), XeqY(Rel1→end,Req9→end), XeqY(Rel1→end,Req10→end), XeqY(Rel1→end,Req11→end), XeqY(Rel1→end,Req12→end), XeqY(Rel1→end,Req13→end), XeqY(Rel1→end,Req14→end), XeqY(Rel1→end,Req15→end), XeqY(Rel1→end,Req16→end), XeqY(Rel1→end,Req17→end), XeqY(Rel1→end,Req18→end), XeqY(Rel1→end,Req19→end), XeqY(Rel1→end,Req20→end), XeqY(Rel1→end,Req21→end), XeqY(Rel1→end,Req22→end), XeqY(Rel1→end,Req23→end), XeqY(Rel1→end,Req24→end), XeqY(Rel1→end,Req25→end), XeqY(Rel1→end,Req26→end), XeqY(Rel1→end,Req27→end), XeqY(Rel1→end,Req28→end), XeqY(Rel1→end,Req29→end), XeqY(Rel1→end,Req30→end), XeqY(Rel1→end,Req31→end), XeqY(Rel1→end,Req32→end), XeqY(Rel1→end,Req33→end), XeqY(Rel1→end,Req34→end), XeqY(Rel1→end,Req35→end), XeqY(Rel1→end,Req36→end), XeqY(Rel1→end,Req37→end), XeqY(Rel1→end,Req38→end), XeqY(Rel1→end,Req39→end), XeqY(Rel1→end,Req40→end), XeqY(Rel1→end,Req41→end), XeqY(Rel1→end,Req42→end), XeqY(Rel1→end,Req43→end), XeqY(Rel1→end,Req44→end), XeqY(Rel1→end,Req45→end), XeqY(Rel1→end,Req46→end), XeqY(Rel1→end,Req47→end), XeqY(Rel1→end,Req48→end), XeqC(Rel 1→end,66))
Possible ends of Release3	:	Or (XeqY(Rel2→end,Req0→end), XeqY(Rel2→end,Req1→end), XeqY(Rel2→end,Req2→end), XeqY(Rel2→end,Req3→end), XeqY(Rel2→end,Req4→end), XeqY(Rel2→end,Req5→end), XeqY(Rel2→end,Req6→end), XeqY(Rel2→end,Req7→end), XeqY(Rel2→end,Req8→end), XeqY(Rel2→end,Req9→end), XeqY(Rel2→end,Req10→end), XeqY(Rel2→end,Req11→end), XeqY(Rel2→end,Req12→end), XeqY(Rel2→end,Req13→end), XeqY(Rel2→end,Req14→end), XeqY(Rel2→end,Req15→end), XeqY(Rel2→end,Req16→end), XeqY(Rel2→end,Req17→end), XeqY(Rel2→end,Req18→end), XeqY(Rel2→end,Req19→end), XeqY(Rel2→end,Req20→end), XeqY(Rel2→end,Req21→end), XeqY(Rel2→end,Req22→end), XeqY(Rel2→end,Req23→end), XeqY(Rel2→end,Req24→end), XeqY(Rel2→end,Req25→end), XeqY(Rel2→end,Req26→end), XeqY(Rel2→end,Req27→end), XeqY(Rel2→end,Req28→end), XeqY(Rel2→end,Req29→end), XeqY(Rel2→end,Req30→end), XeqY(Rel2→end,Req31→end), XeqY(Rel2→end,Req32→end), XeqY(Rel2→end,Req33→end), XeqY(Rel2→end,Req34→end), XeqY(Rel2→end,Req35→end), XeqY(Rel2→end,Req36→end), XeqY(Rel2→end,Req37→end), XeqY(Rel2→end,Req38→end), XeqY(Rel2→end,Req39→end), XeqY(Rel2→end,Req40→end), XeqY(Rel2→end,Req41→end), XeqY(Rel2→end,Req42→end), XeqY(Rel2→end,Req43→end), XeqY(Rel2→end,Req44→end), XeqY(Rel2→end,Req45→end), XeqY(Rel2→end,Req46→end), XeqY(Rel2→end,Req47→end), XeqY(Rel2→end,Req48→end), XeqC(Rel 2→end,95))

Table B.4: Example Application: Start and end constraints of releases.

Constraints for requirement implementation starts

Possible starts of Multilingualism: Or(
XeqY(Req0→start,Req1→end), XeqY(Req0→start,Req2→end), XeqY(Req0→start,Req3→end), XeqY(Req0→start,Req4→end),
XeqY(Req0→start,Req5→end), XeqY(Req0→start,Req6→end), XeqY(Req0→start,Req7→end), XeqY(Req0→start,Req8→end),
XeqY(Req0→start,Req9→end), XeqY(Req0→start,Req10→end), XeqY(Req0→start,Req11→end), XeqY(Req0→start,Req12→end),
XeqY(Req0→start,Req13→end), XeqY(Req0→start,Req14→end), XeqY(Req0→start,Req15→end), XeqY(Req0→start,Req16→end),
XeqY(Req0→start,Req17→end), XeqY(Req0→start,Req18→end), XeqY(Req0→start,Req19→end), XeqY(Req0→start,Req20→end),
XeqY(Req0→start,Req21→end), XeqY(Req0→start,Req22→end), XeqY(Req0→start,Req23→end), XeqY(Req0→start,Req24→end),
XeqY(Req0→start,Req25→end), XeqY(Req0→start,Req26→end), XeqY(Req0→start,Req27→end), XeqY(Req0→start,Req28→end),
XeqY(Req0→start,Req29→end), XeqY(Req0→start,Req30→end), XeqY(Req0→start,Req31→end), XeqY(Req0→start,Req32→end),
XeqY(Req0→start,Req33→end), XeqY(Req0→start,Req34→end), XeqY(Req0→start,Req35→end), XeqY(Req0→start,Req36→end),
XeqY(Req0→start,Req37→end), XeqY(Req0→start,Req38→end), XeqY(Req0→start,Req39→end), XeqY(Req0→start,Req40→end),
XeqY(Req0→start,Req41→end), XeqY(Req0→start,Req42→end), XeqY(Req0→start,Req43→end), XeqY(Req0→start,Req44→end),
XeqY(Req0→start,Req45→end), XeqY(Req0→start,Req46→end), XeqY(Req0→start,Req47→end), XeqY(Req0→start,Req48→end),
XeqY(Req0→start,Rel0→start), XeqY(Req0→start,Rel1→start), XeqY(Req0→start,Rel2→start), XeqY(Req0→start, project_end)))

Possible starts of Authentication: Or(
XeqY(Req1→start,Req0→end), XeqY(Req1→start,Req2→end), XeqY(Req1→start,Req3→end), XeqY(Req1→start,Req4→end),
XeqY(Req1→start,Req5→end), XeqY(Req1→start,Req6→end), XeqY(Req1→start,Req7→end), XeqY(Req1→start,Req8→end),
XeqY(Req1→start,Req9→end), XeqY(Req1→start,Req10→end), XeqY(Req1→start,Req11→end), XeqY(Req1→start,Req12→end),
XeqY(Req1→start,Req13→end), XeqY(Req1→start,Req14→end), XeqY(Req1→start,Req15→end), XeqY(Req1→start,Req16→end),
XeqY(Req1→start,Req17→end), XeqY(Req1→start,Req18→end), XeqY(Req1→start,Req19→end), XeqY(Req1→start,Req20→end),
XeqY(Req1→start,Req21→end), XeqY(Req1→start,Req22→end), XeqY(Req1→start,Req23→end), XeqY(Req1→start,Req24→end),
XeqY(Req1→start,Req25→end), XeqY(Req1→start,Req26→end), XeqY(Req1→start,Req27→end), XeqY(Req1→start,Req28→end),
XeqY(Req1→start,Req29→end), XeqY(Req1→start,Req30→end), XeqY(Req1→start,Req31→end), XeqY(Req1→start,Req32→end),
XeqY(Req1→start,Req33→end), XeqY(Req1→start,Req34→end), XeqY(Req1→start,Req35→end), XeqY(Req1→start,Req36→end),
XeqY(Req1→start,Req37→end), XeqY(Req1→start,Req38→end), XeqY(Req1→start,Req39→end), XeqY(Req1→start,Req40→end),
XeqY(Req1→start,Req41→end), XeqY(Req1→start,Req42→end), XeqY(Req1→start,Req43→end), XeqY(Req1→start,Req44→end),
XeqY(Req1→start,Req45→end), XeqY(Req1→start,Req46→end), XeqY(Req1→start,Req47→end), XeqY(Req1→start,Req48→end),
XeqY(Req1→start,Rel0→start), XeqY(Req1→start,Rel1→start), XeqY(Req1→start,Rel2→start), XeqY(Req1→start, project_end)))

Possible starts of Collectors' surface: Or(
XeqY(Req2→start,Req0→end), XeqY(Req2→start,Req1→end), XeqY(Req2→start,Req3→end), XeqY(Req2→start,Req4→end),
XeqY(Req2→start,Req5→end), XeqY(Req2→start,Req6→end), XeqY(Req2→start,Req7→end), XeqY(Req2→start,Req8→end),
XeqY(Req2→start,Req9→end), XeqY(Req2→start,Req10→end), XeqY(Req2→start,Req11→end), XeqY(Req2→start,Req12→end),
XeqY(Req2→start,Req13→end), XeqY(Req2→start,Req14→end), XeqY(Req2→start,Req15→end), XeqY(Req2→start,Req16→end),
XeqY(Req2→start,Req17→end), XeqY(Req2→start,Req18→end), XeqY(Req2→start,Req19→end), XeqY(Req2→start,Req20→end),
XeqY(Req2→start,Req21→end), XeqY(Req2→start,Req22→end), XeqY(Req2→start,Req23→end), XeqY(Req2→start,Req24→end),
XeqY(Req2→start,Req25→end), XeqY(Req2→start,Req26→end), XeqY(Req2→start,Req27→end), XeqY(Req2→start,Req28→end),
XeqY(Req2→start,Req29→end), XeqY(Req2→start,Req30→end), XeqY(Req2→start,Req31→end), XeqY(Req2→start,Req32→end),
XeqY(Req2→start,Req33→end), XeqY(Req2→start,Req34→end), XeqY(Req2→start,Req35→end), XeqY(Req2→start,Req36→end),
XeqY(Req2→start,Req37→end), XeqY(Req2→start,Req38→end), XeqY(Req2→start,Req39→end), XeqY(Req2→start,Req40→end),
XeqY(Req2→start,Req41→end), XeqY(Req2→start,Req42→end), XeqY(Req2→start,Req43→end), XeqY(Req2→start,Req44→end),
XeqY(Req2→start,Req45→end), XeqY(Req2→start,Req46→end), XeqY(Req2→start,Req47→end), XeqY(Req2→start,Req48→end),
XeqY(Req2→start,Rel0→start), XeqY(Req2→start,Rel1→start), XeqY(Req2→start,Rel2→start), XeqY(Req2→start, project_end)))

Possible starts of GUI Preconditions: Or(
XeqY(Req3→start,Req0→end), XeqY(Req3→start,Req1→end), XeqY(Req3→start,Req2→end), XeqY(Req3→start,Req4→end),
XeqY(Req3→start,Req5→end), XeqY(Req3→start,Req6→end), XeqY(Req3→start,Req7→end), XeqY(Req3→start,Req8→end),
XeqY(Req3→start,Req9→end), XeqY(Req3→start,Req10→end), XeqY(Req3→start,Req11→end), XeqY(Req3→start,Req12→end),
XeqY(Req3→start,Req13→end), XeqY(Req3→start,Req14→end), XeqY(Req3→start,Req15→end), XeqY(Req3→start,Req16→end),
XeqY(Req3→start,Req17→end), XeqY(Req3→start,Req18→end), XeqY(Req3→start,Req19→end), XeqY(Req3→start,Req20→end),
XeqY(Req3→start,Req21→end), XeqY(Req3→start,Req22→end), XeqY(Req3→start,Req23→end), XeqY(Req3→start,Req24→end),
XeqY(Req3→start,Req25→end), XeqY(Req3→start,Req26→end), XeqY(Req3→start,Req27→end), XeqY(Req3→start,Req28→end),
XeqY(Req3→start,Req29→end), XeqY(Req3→start,Req30→end), XeqY(Req3→start,Req31→end), XeqY(Req3→start,Req32→end),
XeqY(Req3→start,Req33→end), XeqY(Req3→start,Req34→end), XeqY(Req3→start,Req35→end), XeqY(Req3→start,Req36→end),
XeqY(Req3→start,Req37→end), XeqY(Req3→start,Req38→end), XeqY(Req3→start,Req39→end), XeqY(Req3→start,Req40→end),
XeqY(Req3→start,Req41→end), XeqY(Req3→start,Req42→end), XeqY(Req3→start,Req43→end), XeqY(Req3→start,Req44→end),
XeqY(Req3→start,Req45→end), XeqY(Req3→start,Req46→end), XeqY(Req3→start,Req47→end), XeqY(Req3→start,Req48→end),
XeqY(Req3→start,Rel0→start), XeqY(Req3→start,Rel1→start), XeqY(Req3→start,Rel2→start), XeqY(Req3→start, project_end)))

Possible starts of Collectors' surface (GUI): Or(
XeqY(Req4→start,Req0→end), XeqY(Req4→start,Req1→end), XeqY(Req4→start,Req2→end), XeqY(Req4→start,Req3→end),
XeqY(Req4→start,Req5→end), XeqY(Req4→start,Req6→end), XeqY(Req4→start,Req7→end), XeqY(Req4→start,Req8→end),
XeqY(Req4→start,Req9→end), XeqY(Req4→start,Req10→end), XeqY(Req4→start,Req11→end), XeqY(Req4→start,Req12→end),
XeqY(Req4→start,Req13→end), XeqY(Req4→start,Req14→end), XeqY(Req4→start,Req15→end), XeqY(Req4→start,Req16→end),
XeqY(Req4→start,Req17→end), XeqY(Req4→start,Req18→end), XeqY(Req4→start,Req19→end), XeqY(Req4→start,Req20→end),
XeqY(Req4→start,Req21→end), XeqY(Req4→start,Req22→end), XeqY(Req4→start,Req23→end), XeqY(Req4→start,Req24→end),
XeqY(Req4→start,Req25→end), XeqY(Req4→start,Req26→end), XeqY(Req4→start,Req27→end), XeqY(Req4→start,Req28→end),
XeqY(Req4→start,Req29→end), XeqY(Req4→start,Req30→end), XeqY(Req4→start,Req31→end), XeqY(Req4→start,Req32→end),
XeqY(Req4→start,Req33→end), XeqY(Req4→start,Req34→end), XeqY(Req4→start,Req35→end), XeqY(Req4→start,Req36→end),
XeqY(Req4→start,Req37→end), XeqY(Req4→start,Req38→end), XeqY(Req4→start,Req39→end), XeqY(Req4→start,Req40→end),
XeqY(Req4→start,Req41→end), XeqY(Req4→start,Req42→end), XeqY(Req4→start,Req43→end), XeqY(Req4→start,Req44→end),
XeqY(Req4→start,Req45→end), XeqY(Req4→start,Req46→end), XeqY(Req4→start,Req47→end), XeqY(Req4→start,Req48→end),
XeqY(Req4→start,Rel0→start), XeqY(Req4→start,Rel1→start), XeqY(Req4→start,Rel2→start), XeqY(Req4→start, project_end)))

Possible starts of Database Backend: Or(
XeqY(Req5→start,Req0→end), XeqY(Req5→start,Req1→end), XeqY(Req5→start,Req2→end), XeqY(Req5→start,Req3→end),
XeqY(Req5→start,Req4→end), XeqY(Req5→start,Req6→end), XeqY(Req5→start,Req7→end), XeqY(Req5→start,Req8→end),
XeqY(Req5→start,Req9→end), XeqY(Req5→start,Req10→end), XeqY(Req5→start,Req11→end), XeqY(Req5→start,Req12→end),
XeqY(Req5→start,Req13→end), XeqY(Req5→start,Req14→end), XeqY(Req5→start,Req15→end), XeqY(Req5→start,Req16→end),
XeqY(Req5→start,Req17→end), XeqY(Req5→start,Req18→end), XeqY(Req5→start,Req19→end), XeqY(Req5→start,Req20→end),
XeqY(Req5→start,Req21→end), XeqY(Req5→start,Req22→end), XeqY(Req5→start,Req23→end), XeqY(Req5→start,Req24→end),
XeqY(Req5→start,Req25→end), XeqY(Req5→start,Req26→end), XeqY(Req5→start,Req27→end), XeqY(Req5→start,Req28→end),
XeqY(Req5→start,Req29→end), XeqY(Req5→start,Req30→end), XeqY(Req5→start,Req31→end), XeqY(Req5→start,Req32→end),
XeqY(Req5→start,Req33→end), XeqY(Req5→start,Req34→end), XeqY(Req5→start,Req35→end), XeqY(Req5→start,Req36→end),
XeqY(Req5→start,Req37→end), XeqY(Req5→start,Req38→end), XeqY(Req5→start,Req39→end), XeqY(Req5→start,Req40→end),
XeqY(Req5→start,Req41→end), XeqY(Req5→start,Req42→end), XeqY(Req5→start,Req43→end), XeqY(Req5→start,Req44→end),
XeqY(Req5→start,Req45→end), XeqY(Req5→start,Req46→end), XeqY(Req5→start,Req47→end), XeqY(Req5→start,Req48→end),
XeqY(Req5→start,Rel0→start), XeqY(Req5→start,Rel1→start), XeqY(Req5→start,Rel2→start), XeqY(Req5→start, project_end)))

Possible starts of Collectors and installation: Or(
XeqY(Req6→start,Req0→end), XeqY(Req6→start,Req1→end), XeqY(Req6→start,Req2→end), XeqY(Req6→start,Req3→end),
XeqY(Req6→start,Req4→end), XeqY(Req6→start,Req5→end), XeqY(Req6→start,Req7→end), XeqY(Req6→start,Req8→end),
XeqY(Req6→start,Req9→end), XeqY(Req6→start,Req10→end), XeqY(Req6→start,Req11→end), XeqY(Req6→start,Req12→end),
XeqY(Req6→start,Req13→end), XeqY(Req6→start,Req14→end), XeqY(Req6→start,Req15→end), XeqY(Req6→start,Req16→end),
XeqY(Req6→start,Req17→end), XeqY(Req6→start,Req18→end), XeqY(Req6→start,Req19→end), XeqY(Req6→start,Req20→end),
XeqY(Req6→start,Req21→end), XeqY(Req6→start,Req22→end), XeqY(Req6→start,Req23→end), XeqY(Req6→start,Req24→end),
XeqY(Req6→start,Req25→end), XeqY(Req6→start,Req26→end), XeqY(Req6→start,Req27→end), XeqY(Req6→start,Req28→end),
XeqY(Req6→start,Req29→end), XeqY(Req6→start,Req30→end), XeqY(Req6→start,Req31→end), XeqY(Req6→start,Req32→end),
XeqY(Req6→start,Req33→end), XeqY(Req6→start,Req34→end), XeqY(Req6→start,Req35→end), XeqY(Req6→start,Req36→end),
XeqY(Req6→start,Req37→end), XeqY(Req6→start,Req38→end), XeqY(Req6→start,Req39→end), XeqY(Req6→start,Req40→end),
XeqY(Req6→start,Req41→end), XeqY(Req6→start,Req42→end), XeqY(Req6→start,Req43→end), XeqY(Req6→start,Req44→end),
XeqY(Req6→start,Req45→end), XeqY(Req6→start,Req46→end), XeqY(Req6→start,Req47→end), XeqY(Req6→start,Req48→end),
XeqY(Req6→start,Rel0→start), XeqY(Req6→start,Rel1→start), XeqY(Req6→start,Rel2→start), XeqY(Req6→start, project_end)))

Possible starts of Collectors's surface (Store): Or(
XeqY(Req7→start,Req0→end), XeqY(Req7→start,Req1→end), XeqY(Req7→start,Req2→end), XeqY(Req7→start,Req3→end),
XeqY(Req7→start,Req4→end), XeqY(Req7→start,Req5→end), XeqY(Req7→start,Req6→end), XeqY(Req7→start,Req8→end),
XeqY(Req7→start,Req9→end), XeqY(Req7→start,Req10→end), XeqY(Req7→start,Req11→end), XeqY(Req7→start,Req12→end),
XeqY(Req7→start,Req13→end), XeqY(Req7→start,Req14→end), XeqY(Req7→start,Req15→end), XeqY(Req7→start,Req16→end),
XeqY(Req7→start,Req17→end), XeqY(Req7→start,Req18→end), XeqY(Req7→start,Req19→end), XeqY(Req7→start,Req20→end),
XeqY(Req7→start,Req21→end), XeqY(Req7→start,Req22→end), XeqY(Req7→start,Req23→end), XeqY(Req7→start,Req24→end),
XeqY(Req7→start,Req25→end), XeqY(Req7→start,Req26→end), XeqY(Req7→start,Req27→end), XeqY(Req7→start,Req28→end),
XeqY(Req7→start,Req29→end), XeqY(Req7→start,Req30→end), XeqY(Req7→start,Req31→end), XeqY(Req7→start,Req32→end),
XeqY(Req7→start,Req33→end), XeqY(Req7→start,Req34→end), XeqY(Req7→start,Req35→end), XeqY(Req7→start,Req36→end),
XeqY(Req7→start,Req37→end), XeqY(Req7→start,Req38→end), XeqY(Req7→start,Req39→end), XeqY(Req7→start,Req40→end),
XeqY(Req7→start,Req41→end), XeqY(Req7→start,Req42→end), XeqY(Req7→start,Req43→end), XeqY(Req7→start,Req44→end),
XeqY(Req7→start,Req45→end), XeqY(Req7→start,Req46→end), XeqY(Req7→start,Req47→end), XeqY(Req7→start,Req48→end),
XeqY(Req7→start,Rel0→start), XeqY(Req7→start,Rel1→start), XeqY(Req7→start,Rel2→start), XeqY(Req7→start, project_end)))

Possible starts of Collectors and installation (Store): Or(
XeqY(Req8→start,Req0→end), XeqY(Req8→start,Req1→end), XeqY(Req8→start,Req2→end), XeqY(Req8→start,Req3→end),
XeqY(Req8→start,Req4→end), XeqY(Req8→start,Req5→end), XeqY(Req8→start,Req6→end), XeqY(Req8→start,Req7→end),
XeqY(Req8→start,Req9→end), XeqY(Req8→start,Req10→end), XeqY(Req8→start,Req11→end), XeqY(Req8→start,Req12→end),
XeqY(Req8→start,Req13→end), XeqY(Req8→start,Req14→end), XeqY(Req8→start,Req15→end), XeqY(Req8→start,Req16→end),
XeqY(Req8→start,Req17→end), XeqY(Req8→start,Req18→end), XeqY(Req8→start,Req19→end), XeqY(Req8→start,Req20→end),
XeqY(Req8→start,Req21→end), XeqY(Req8→start,Req22→end), XeqY(Req8→start,Req23→end), XeqY(Req8→start,Req24→end),
XeqY(Req8→start,Req25→end), XeqY(Req8→start,Req26→end), XeqY(Req8→start,Req27→end), XeqY(Req8→start,Req28→end),
XeqY(Req8→start,Req29→end), XeqY(Req8→start,Req30→end), XeqY(Req8→start,Req31→end), XeqY(Req8→start,Req32→end),
XeqY(Req8→start,Req33→end), XeqY(Req8→start,Req34→end), XeqY(Req8→start,Req35→end), XeqY(Req8→start,Req36→end),
XeqY(Req8→start,Req37→end), XeqY(Req8→start,Req38→end), XeqY(Req8→start,Req39→end), XeqY(Req8→start,Req40→end),
XeqY(Req8→start,Req41→end), XeqY(Req8→start,Req42→end), XeqY(Req8→start,Req43→end), XeqY(Req8→start,Req44→end),
XeqY(Req8→start,Req45→end), XeqY(Req8→start,Req46→end), XeqY(Req8→start,Req47→end), XeqY(Req8→start,Req48→end),
XeqY(Req8→start,Rel0→start), XeqY(Req8→start,Rel1→start), XeqY(Req8→start,Rel2→start), XeqY(Req8→start, project_end)))

Possible starts of Air caps calculation: Or(
XeqY(Req9→start,Req0→end), XeqY(Req9→start,Req1→end), XeqY(Req9→start,Req2→end), XeqY(Req9→start,Req3→end),
XeqY(Req9→start,Req4→end), XeqY(Req9→start,Req5→end), XeqY(Req9→start,Req6→end), XeqY(Req9→start,Req7→end),
XeqY(Req9→start,Req8→end), XeqY(Req9→start,Req10→end), XeqY(Req9→start,Req11→end), XeqY(Req9→start,Req12→end),
XeqY(Req9→start,Req13→end), XeqY(Req9→start,Req14→end), XeqY(Req9→start,Req15→end), XeqY(Req9→start,Req16→end),
XeqY(Req9→start,Req17→end), XeqY(Req9→start,Req18→end), XeqY(Req9→start,Req19→end), XeqY(Req9→start,Req20→end),
XeqY(Req9→start,Req21→end), XeqY(Req9→start,Req22→end), XeqY(Req9→start,Req23→end), XeqY(Req9→start,Req24→end),
XeqY(Req9→start,Req25→end), XeqY(Req9→start,Req26→end), XeqY(Req9→start,Req27→end), XeqY(Req9→start,Req28→end),
XeqY(Req9→start,Req29→end), XeqY(Req9→start,Req30→end), XeqY(Req9→start,Req31→end), XeqY(Req9→start,Req32→end),
XeqY(Req9→start,Req33→end), XeqY(Req9→start,Req34→end), XeqY(Req9→start,Req35→end), XeqY(Req9→start,Req36→end),
XeqY(Req9→start,Req37→end), XeqY(Req9→start,Req38→end), XeqY(Req9→start,Req39→end), XeqY(Req9→start,Req40→end),
XeqY(Req9→start,Req41→end), XeqY(Req9→start,Req42→end), XeqY(Req9→start,Req43→end), XeqY(Req9→start,Req44→end),
XeqY(Req9→start,Req45→end), XeqY(Req9→start,Req46→end), XeqY(Req9→start,Req47→end), XeqY(Req9→start,Req48→end),
XeqY(Req9→start,Rel0→start), XeqY(Req9→start,Rel1→start), XeqY(Req9→start,Rel2→start), XeqY(Req9→start, project_end)))

Possible starts of Air caps calculation (Store): Or(
XeqY(Req10→start,Req0→end), XeqY(Req10→start,Req1→end), XeqY(Req10→start,Req2→end), XeqY(Req10→start,Req3→end),
XeqY(Req10→start,Req4→end), XeqY(Req10→start,Req5→end), XeqY(Req10→start,Req6→end), XeqY(Req10→start,Req7→end),
XeqY(Req10→start,Req8→end), XeqY(Req10→start,Req9→end), XeqY(Req10→start,Req11→end), XeqY(Req10→start,Req12→end),
XeqY(Req10→start,Req13→end), XeqY(Req10→start,Req14→end), XeqY(Req10→start,Req15→end), XeqY(Req10→start,Req16→end),
XeqY(Req10→start,Req17→end), XeqY(Req10→start,Req18→end), XeqY(Req10→start,Req19→end), XeqY(Req10→start,Req20→end),
XeqY(Req10→start,Req21→end), XeqY(Req10→start,Req22→end), XeqY(Req10→start,Req23→end), XeqY(Req10→start,Req24→end),
XeqY(Req10→start,Req25→end), XeqY(Req10→start,Req26→end), XeqY(Req10→start,Req27→end), XeqY(Req10→start,Req28→end),
XeqY(Req10→start,Req29→end), XeqY(Req10→start,Req30→end), XeqY(Req10→start,Req31→end), XeqY(Req10→start,Req32→end),
XeqY(Req10→start,Req33→end), XeqY(Req10→start,Req34→end), XeqY(Req10→start,Req35→end), XeqY(Req10→start,Req36→end),
XeqY(Req10→start,Req37→end), XeqY(Req10→start,Req38→end), XeqY(Req10→start,Req39→end), XeqY(Req10→start,Req40→end),
XeqY(Req10→start,Req41→end), XeqY(Req10→start,Req42→end), XeqY(Req10→start,Req43→end), XeqY(Req10→start,Req44→end),
XeqY(Req10→start,Req45→end), XeqY(Req10→start,Req46→end), XeqY(Req10→start,Req47→end), XeqY(Req10→start,Req48→end),
XeqY(Req10→start,Rel0→start), XeqY(Req10→start,Rel1→start), XeqY(Req10→start,Rel2→start), XeqY(Req10→start, project_end)))

Possible starts of Pipe Costs: Or(
XeqY(Req11→start,Req0→end), XeqY(Req11→start,Req1→end), XeqY(Req11→start,Req2→end), XeqY(Req11→start,Req3→end),
XeqY(Req11→start,Req4→end), XeqY(Req11→start,Req5→end), XeqY(Req11→start,Req6→end), XeqY(Req11→start,Req7→end),
XeqY(Req11→start,Req8→end), XeqY(Req11→start,Req9→end), XeqY(Req11→start,Req10→end), XeqY(Req11→start,Req12→end),
XeqY(Req11→start,Req13→end), XeqY(Req11→start,Req14→end), XeqY(Req11→start,Req15→end), XeqY(Req11→start,Req16→end),
XeqY(Req11→start,Req17→end), XeqY(Req11→start,Req18→end), XeqY(Req11→start,Req19→end), XeqY(Req11→start,Req20→end),
XeqY(Req11→start,Req21→end), XeqY(Req11→start,Req22→end), XeqY(Req11→start,Req23→end), XeqY(Req11→start,Req24→end),
XeqY(Req11→start,Req25→end), XeqY(Req11→start,Req26→end), XeqY(Req11→start,Req27→end), XeqY(Req11→start,Req28→end),
XeqY(Req11→start,Req29→end), XeqY(Req11→start,Req30→end), XeqY(Req11→start,Req31→end), XeqY(Req11→start,Req32→end),
XeqY(Req11→start,Req33→end), XeqY(Req11→start,Req34→end), XeqY(Req11→start,Req35→end), XeqY(Req11→start,Req36→end),
XeqY(Req11→start,Req37→end), XeqY(Req11→start,Req38→end), XeqY(Req11→start,Req39→end), XeqY(Req11→start,Req40→end),
XeqY(Req11→start,Req41→end), XeqY(Req11→start,Req42→end), XeqY(Req11→start,Req43→end), XeqY(Req11→start,Req44→end),
XeqY(Req11→start,Req45→end), XeqY(Req11→start,Req46→end), XeqY(Req11→start,Req47→end), XeqY(Req11→start,Req48→end),
XeqY(Req11→start,Rel0→start), XeqY(Req11→start,Rel1→start), XeqY(Req11→start,Rel2→start), XeqY(Req11→start, project_end)))

Possible starts of Pipe Costs (Store): Or(
XeqY(Req12→start,Req0→end), XeqY(Req12→start,Req1→end), XeqY(Req12→start,Req2→end), XeqY(Req12→start,Req3→end),
XeqY(Req12→start,Req4→end), XeqY(Req12→start,Req5→end), XeqY(Req12→start,Req6→end), XeqY(Req12→start,Req7→end),
XeqY(Req12→start,Req8→end), XeqY(Req12→start,Req9→end), XeqY(Req12→start,Req10→end), XeqY(Req12→start,Req11→end),
XeqY(Req12→start,Req13→end), XeqY(Req12→start,Req14→end), XeqY(Req12→start,Req15→end), XeqY(Req12→start,Req16→end),
XeqY(Req12→start,Req17→end), XeqY(Req12→start,Req18→end), XeqY(Req12→start,Req19→end), XeqY(Req12→start,Req20→end),
XeqY(Req12→start,Req21→end), XeqY(Req12→start,Req22→end), XeqY(Req12→start,Req23→end), XeqY(Req12→start,Req24→end),
XeqY(Req12→start,Req25→end), XeqY(Req12→start,Req26→end), XeqY(Req12→start,Req27→end), XeqY(Req12→start,Req28→end),
XeqY(Req12→start,Req29→end), XeqY(Req12→start,Req30→end), XeqY(Req12→start,Req31→end), XeqY(Req12→start,Req32→end),
XeqY(Req12→start,Req33→end), XeqY(Req12→start,Req34→end), XeqY(Req12→start,Req35→end), XeqY(Req12→start,Req36→end),
XeqY(Req12→start,Req37→end), XeqY(Req12→start,Req38→end), XeqY(Req12→start,Req39→end), XeqY(Req12→start,Req40→end),
XeqY(Req12→start,Req41→end), XeqY(Req12→start,Req42→end), XeqY(Req12→start,Req43→end), XeqY(Req12→start,Req44→end),
XeqY(Req12→start,Req45→end), XeqY(Req12→start,Req46→end), XeqY(Req12→start,Req47→end), XeqY(Req12→start,Req48→end),
XeqY(Req12→start,Rel0→start), XeqY(Req12→start,Rel1→start), XeqY(Req12→start,Rel2→start), XeqY(Req12→start, project_end)))

Possible starts of PGR & ADG: Or(
XeqY(Req13→start,Req0→end), XeqY(Req13→start,Req1→end), XeqY(Req13→start,Req2→end), XeqY(Req13→start,Req3→end),
XeqY(Req13→start,Req4→end), XeqY(Req13→start,Req5→end), XeqY(Req13→start,Req6→end), XeqY(Req13→start,Req7→end),
XeqY(Req13→start,Req8→end), XeqY(Req13→start,Req9→end), XeqY(Req13→start,Req10→end), XeqY(Req13→start,Req11→end),
XeqY(Req13→start,Req12→end), XeqY(Req13→start,Req14→end), XeqY(Req13→start,Req15→end), XeqY(Req13→start,Req16→end),
XeqY(Req13→start,Req17→end), XeqY(Req13→start,Req18→end), XeqY(Req13→start,Req19→end), XeqY(Req13→start,Req20→end),
XeqY(Req13→start,Req21→end), XeqY(Req13→start,Req22→end), XeqY(Req13→start,Req23→end), XeqY(Req13→start,Req24→end),
XeqY(Req13→start,Req25→end), XeqY(Req13→start,Req26→end), XeqY(Req13→start,Req27→end), XeqY(Req13→start,Req28→end),
XeqY(Req13→start,Req29→end), XeqY(Req13→start,Req30→end), XeqY(Req13→start,Req31→end), XeqY(Req13→start,Req32→end),
XeqY(Req13→start,Req33→end), XeqY(Req13→start,Req34→end), XeqY(Req13→start,Req35→end), XeqY(Req13→start,Req36→end),
XeqY(Req13→start,Req37→end), XeqY(Req13→start,Req38→end), XeqY(Req13→start,Req39→end), XeqY(Req13→start,Req40→end),
XeqY(Req13→start,Req41→end), XeqY(Req13→start,Req42→end), XeqY(Req13→start,Req43→end), XeqY(Req13→start,Req44→end),
XeqY(Req13→start,Req45→end), XeqY(Req13→start,Req46→end), XeqY(Req13→start,Req47→end), XeqY(Req13→start,Req48→end),
XeqY(Req13→start,Rel0→start), XeqY(Req13→start,Rel1→start), XeqY(Req13→start,Rel2→start), XeqY(Req13→start, project_end)))

Possible starts of PGR & ADG (Store): Or(
XeqY(Req14→start,Req0→end), XeqY(Req14→start,Req1→end), XeqY(Req14→start,Req2→end), XeqY(Req14→start,Req3→end),
XeqY(Req14→start,Req4→end), XeqY(Req14→start,Req5→end), XeqY(Req14→start,Req6→end), XeqY(Req14→start,Req7→end),
XeqY(Req14→start,Req8→end), XeqY(Req14→start,Req9→end), XeqY(Req14→start,Req10→end), XeqY(Req14→start,Req11→end),
XeqY(Req14→start,Req12→end), XeqY(Req14→start,Req13→end), XeqY(Req14→start,Req15→end), XeqY(Req14→start,Req16→end),
XeqY(Req14→start,Req17→end), XeqY(Req14→start,Req18→end), XeqY(Req14→start,Req19→end), XeqY(Req14→start,Req20→end),
XeqY(Req14→start,Req21→end), XeqY(Req14→start,Req22→end), XeqY(Req14→start,Req23→end), XeqY(Req14→start,Req24→end),
XeqY(Req14→start,Req25→end), XeqY(Req14→start,Req26→end), XeqY(Req14→start,Req27→end), XeqY(Req14→start,Req28→end),
XeqY(Req14→start,Req29→end), XeqY(Req14→start,Req30→end), XeqY(Req14→start,Req31→end), XeqY(Req14→start,Req32→end),
XeqY(Req14→start,Req33→end), XeqY(Req14→start,Req34→end), XeqY(Req14→start,Req35→end), XeqY(Req14→start,Req36→end),
XeqY(Req14→start,Req37→end), XeqY(Req14→start,Req38→end), XeqY(Req14→start,Req39→end), XeqY(Req14→start,Req40→end),
XeqY(Req14→start,Req41→end), XeqY(Req14→start,Req42→end), XeqY(Req14→start,Req43→end), XeqY(Req14→start,Req44→end),
XeqY(Req14→start,Req45→end), XeqY(Req14→start,Req46→end), XeqY(Req14→start,Req47→end), XeqY(Req14→start,Req48→end),
XeqY(Req14→start,Rel0→start), XeqY(Req14→start,Rel1→start), XeqY(Req14→start,Rel2→start), XeqY(Req14→start, project_end)))

Possible starts of Costs for refilling: Or(
XeqY(Req15→start,Req0→end), XeqY(Req15→start,Req1→end), XeqY(Req15→start,Req2→end), XeqY(Req15→start,Req3→end),
XeqY(Req15→start,Req4→end), XeqY(Req15→start,Req5→end), XeqY(Req15→start,Req6→end), XeqY(Req15→start,Req7→end),
XeqY(Req15→start,Req8→end), XeqY(Req15→start,Req9→end), XeqY(Req15→start,Req10→end), XeqY(Req15→start,Req11→end),
XeqY(Req15→start,Req12→end), XeqY(Req15→start,Req13→end), XeqY(Req15→start,Req14→end), XeqY(Req15→start,Req16→end),
XeqY(Req15→start,Req17→end), XeqY(Req15→start,Req18→end), XeqY(Req15→start,Req19→end), XeqY(Req15→start,Req20→end),
XeqY(Req15→start,Req21→end), XeqY(Req15→start,Req22→end), XeqY(Req15→start,Req23→end), XeqY(Req15→start,Req24→end),
XeqY(Req15→start,Req25→end), XeqY(Req15→start,Req26→end), XeqY(Req15→start,Req27→end), XeqY(Req15→start,Req28→end),
XeqY(Req15→start,Req29→end), XeqY(Req15→start,Req30→end), XeqY(Req15→start,Req31→end), XeqY(Req15→start,Req32→end),
XeqY(Req15→start,Req33→end), XeqY(Req15→start,Req34→end), XeqY(Req15→start,Req35→end), XeqY(Req15→start,Req36→end),
XeqY(Req15→start,Req37→end), XeqY(Req15→start,Req38→end), XeqY(Req15→start,Req39→end), XeqY(Req15→start,Req40→end),
XeqY(Req15→start,Req41→end), XeqY(Req15→start,Req42→end), XeqY(Req15→start,Req43→end), XeqY(Req15→start,Req44→end),
XeqY(Req15→start,Req45→end), XeqY(Req15→start,Req46→end), XeqY(Req15→start,Req47→end), XeqY(Req15→start,Req48→end),
XeqY(Req15→start,Rel0→start), XeqY(Req15→start,Rel1→start), XeqY(Req15→start,Rel2→start), XeqY(Req15→start, project_end)))

Possible starts of Costs for refilling (Store): Or(
XeqY(Req16→start,Req0→end), XeqY(Req16→start,Req1→end), XeqY(Req16→start,Req2→end), XeqY(Req16→start,Req3→end),
XeqY(Req16→start,Req4→end), XeqY(Req16→start,Req5→end), XeqY(Req16→start,Req6→end), XeqY(Req16→start,Req7→end),
XeqY(Req16→start,Req8→end), XeqY(Req16→start,Req9→end), XeqY(Req16→start,Req10→end), XeqY(Req16→start,Req11→end),
XeqY(Req16→start,Req12→end), XeqY(Req16→start,Req13→end), XeqY(Req16→start,Req14→end), XeqY(Req16→start,Req15→end),
XeqY(Req16→start,Req17→end), XeqY(Req16→start,Req18→end), XeqY(Req16→start,Req19→end), XeqY(Req16→start,Req20→end),
XeqY(Req16→start,Req21→end), XeqY(Req16→start,Req22→end), XeqY(Req16→start,Req23→end), XeqY(Req16→start,Req24→end),
XeqY(Req16→start,Req25→end), XeqY(Req16→start,Req26→end), XeqY(Req16→start,Req27→end), XeqY(Req16→start,Req28→end),
XeqY(Req16→start,Req29→end), XeqY(Req16→start,Req30→end), XeqY(Req16→start,Req31→end), XeqY(Req16→start,Req32→end),
XeqY(Req16→start,Req33→end), XeqY(Req16→start,Req34→end), XeqY(Req16→start,Req35→end), XeqY(Req16→start,Req36→end),
XeqY(Req16→start,Req37→end), XeqY(Req16→start,Req38→end), XeqY(Req16→start,Req39→end), XeqY(Req16→start,Req40→end),
XeqY(Req16→start,Req41→end), XeqY(Req16→start,Req42→end), XeqY(Req16→start,Req43→end), XeqY(Req16→start,Req44→end),
XeqY(Req16→start,Req45→end), XeqY(Req16→start,Req46→end), XeqY(Req16→start,Req47→end), XeqY(Req16→start,Req48→end),
XeqY(Req16→start,Rel0→start), XeqY(Req16→start,Rel1→start), XeqY(Req16→start,Rel2→start), XeqY(Req16→start, project_end)))

Possible starts of Special Equipment: Or(
XeqY(Req17→start,Req0→end), XeqY(Req17→start,Req1→end), XeqY(Req17→start,Req2→end), XeqY(Req17→start,Req3→end),
XeqY(Req17→start,Req4→end), XeqY(Req17→start,Req5→end), XeqY(Req17→start,Req6→end), XeqY(Req17→start,Req7→end),
XeqY(Req17→start,Req8→end), XeqY(Req17→start,Req9→end), XeqY(Req17→start,Req10→end), XeqY(Req17→start,Req11→end),
XeqY(Req17→start,Req12→end), XeqY(Req17→start,Req13→end), XeqY(Req17→start,Req14→end), XeqY(Req17→start,Req15→end),
XeqY(Req17→start,Req16→end), XeqY(Req17→start,Req18→end), XeqY(Req17→start,Req19→end), XeqY(Req17→start,Req20→end),
XeqY(Req17→start,Req21→end), XeqY(Req17→start,Req22→end), XeqY(Req17→start,Req23→end), XeqY(Req17→start,Req24→end),
XeqY(Req17→start,Req25→end), XeqY(Req17→start,Req26→end), XeqY(Req17→start,Req27→end), XeqY(Req17→start,Req28→end),
XeqY(Req17→start,Req29→end), XeqY(Req17→start,Req30→end), XeqY(Req17→start,Req31→end), XeqY(Req17→start,Req32→end),
XeqY(Req17→start,Req33→end), XeqY(Req17→start,Req34→end), XeqY(Req17→start,Req35→end), XeqY(Req17→start,Req36→end),
XeqY(Req17→start,Req37→end), XeqY(Req17→start,Req38→end), XeqY(Req17→start,Req39→end), XeqY(Req17→start,Req40→end),
XeqY(Req17→start,Req41→end), XeqY(Req17→start,Req42→end), XeqY(Req17→start,Req43→end), XeqY(Req17→start,Req44→end),
XeqY(Req17→start,Req45→end), XeqY(Req17→start,Req46→end), XeqY(Req17→start,Req47→end), XeqY(Req17→start,Req48→end),
XeqY(Req17→start,Rel0→start), XeqY(Req17→start,Rel1→start), XeqY(Req17→start,Rel2→start), XeqY(Req17→start, project_end)))

Possible starts of Special Equipment (Store): Or(
XeqY(Req18→start,Req0→end), XeqY(Req18→start,Req1→end), XeqY(Req18→start,Req2→end), XeqY(Req18→start,Req3→end),
XeqY(Req18→start,Req4→end), XeqY(Req18→start,Req5→end), XeqY(Req18→start,Req6→end), XeqY(Req18→start,Req7→end),
XeqY(Req18→start,Req8→end), XeqY(Req18→start,Req9→end), XeqY(Req18→start,Req10→end), XeqY(Req18→start,Req11→end),
XeqY(Req18→start,Req12→end), XeqY(Req18→start,Req13→end), XeqY(Req18→start,Req14→end), XeqY(Req18→start,Req15→end),
XeqY(Req18→start,Req16→end), XeqY(Req18→start,Req17→end), XeqY(Req18→start,Req19→end), XeqY(Req18→start,Req20→end),
XeqY(Req18→start,Req21→end), XeqY(Req18→start,Req22→end), XeqY(Req18→start,Req23→end), XeqY(Req18→start,Req24→end),
XeqY(Req18→start,Req25→end), XeqY(Req18→start,Req26→end), XeqY(Req18→start,Req27→end), XeqY(Req18→start,Req28→end),
XeqY(Req18→start,Req29→end), XeqY(Req18→start,Req30→end), XeqY(Req18→start,Req31→end), XeqY(Req18→start,Req32→end),
XeqY(Req18→start,Req33→end), XeqY(Req18→start,Req34→end), XeqY(Req18→start,Req35→end), XeqY(Req18→start,Req36→end),
XeqY(Req18→start,Req37→end), XeqY(Req18→start,Req38→end), XeqY(Req18→start,Req39→end), XeqY(Req18→start,Req40→end),
XeqY(Req18→start,Req41→end), XeqY(Req18→start,Req42→end), XeqY(Req18→start,Req43→end), XeqY(Req18→start,Req44→end),
XeqY(Req18→start,Req45→end), XeqY(Req18→start,Req46→end), XeqY(Req18→start,Req47→end), XeqY(Req18→start,Req48→end),
XeqY(Req18→start,Rel0→start), XeqY(Req18→start,Rel1→start), XeqY(Req18→start,Rel2→start), XeqY(Req18→start, project_end)))

Possible starts of Misc. Material: Or(
XeqY(Req19→start,Req0→end), XeqY(Req19→start,Req1→end), XeqY(Req19→start,Req2→end), XeqY(Req19→start,Req3→end),
XeqY(Req19→start,Req4→end), XeqY(Req19→start,Req5→end), XeqY(Req19→start,Req6→end), XeqY(Req19→start,Req7→end),
XeqY(Req19→start,Req8→end), XeqY(Req19→start,Req9→end), XeqY(Req19→start,Req10→end), XeqY(Req19→start,Req11→end),
XeqY(Req19→start,Req12→end), XeqY(Req19→start,Req13→end), XeqY(Req19→start,Req14→end), XeqY(Req19→start,Req15→end),
XeqY(Req19→start,Req16→end), XeqY(Req19→start,Req17→end), XeqY(Req19→start,Req18→end), XeqY(Req19→start,Req20→end),
XeqY(Req19→start,Req21→end), XeqY(Req19→start,Req22→end), XeqY(Req19→start,Req23→end), XeqY(Req19→start,Req24→end),
XeqY(Req19→start,Req25→end), XeqY(Req19→start,Req26→end), XeqY(Req19→start,Req27→end), XeqY(Req19→start,Req28→end),
XeqY(Req19→start,Req29→end), XeqY(Req19→start,Req30→end), XeqY(Req19→start,Req31→end), XeqY(Req19→start,Req32→end),
XeqY(Req19→start,Req33→end), XeqY(Req19→start,Req34→end), XeqY(Req19→start,Req35→end), XeqY(Req19→start,Req36→end),
XeqY(Req19→start,Req37→end), XeqY(Req19→start,Req38→end), XeqY(Req19→start,Req39→end), XeqY(Req19→start,Req40→end),
XeqY(Req19→start,Req41→end), XeqY(Req19→start,Req42→end), XeqY(Req19→start,Req43→end), XeqY(Req19→start,Req44→end),
XeqY(Req19→start,Req45→end), XeqY(Req19→start,Req46→end), XeqY(Req19→start,Req47→end), XeqY(Req19→start,Req48→end),
XeqY(Req19→start,Rel0→start), XeqY(Req19→start,Rel1→start), XeqY(Req19→start,Rel2→start), XeqY(Req19→start, project_end)))

Possible starts of Misc. Material (Store): Or(
XeqY(Req20→start,Req0→end), XeqY(Req20→start,Req1→end), XeqY(Req20→start,Req2→end), XeqY(Req20→start,Req3→end),
XeqY(Req20→start,Req4→end), XeqY(Req20→start,Req5→end), XeqY(Req20→start,Req6→end), XeqY(Req20→start,Req7→end),
XeqY(Req20→start,Req8→end), XeqY(Req20→start,Req9→end), XeqY(Req20→start,Req10→end), XeqY(Req20→start,Req11→end),
XeqY(Req20→start,Req12→end), XeqY(Req20→start,Req13→end), XeqY(Req20→start,Req14→end), XeqY(Req20→start,Req15→end),
XeqY(Req20→start,Req16→end), XeqY(Req20→start,Req17→end), XeqY(Req20→start,Req18→end), XeqY(Req20→start,Req19→end),
XeqY(Req20→start,Req21→end), XeqY(Req20→start,Req22→end), XeqY(Req20→start,Req23→end), XeqY(Req20→start,Req24→end),
XeqY(Req20→start,Req25→end), XeqY(Req20→start,Req26→end), XeqY(Req20→start,Req27→end), XeqY(Req20→start,Req28→end),
XeqY(Req20→start,Req29→end), XeqY(Req20→start,Req30→end), XeqY(Req20→start,Req31→end), XeqY(Req20→start,Req32→end),
XeqY(Req20→start,Req33→end), XeqY(Req20→start,Req34→end), XeqY(Req20→start,Req35→end), XeqY(Req20→start,Req36→end),
XeqY(Req20→start,Req37→end), XeqY(Req20→start,Req38→end), XeqY(Req20→start,Req39→end), XeqY(Req20→start,Req40→end),
XeqY(Req20→start,Req41→end), XeqY(Req20→start,Req42→end), XeqY(Req20→start,Req43→end), XeqY(Req20→start,Req44→end),
XeqY(Req20→start,Req45→end), XeqY(Req20→start,Req46→end), XeqY(Req20→start,Req47→end), XeqY(Req20→start,Req48→end),
XeqY(Req20→start,Rel0→start), XeqY(Req20→start,Rel1→start), XeqY(Req20→start,Rel2→start), XeqY(Req20→start, project_end)))

Possible starts of Misc. Material (GUI): Or(
XeqY(Req21→start,Req0→end), XeqY(Req21→start,Req1→end), XeqY(Req21→start,Req2→end), XeqY(Req21→start,Req3→end),
XeqY(Req21→start,Req4→end), XeqY(Req21→start,Req5→end), XeqY(Req21→start,Req6→end), XeqY(Req21→start,Req7→end),
XeqY(Req21→start,Req8→end), XeqY(Req21→start,Req9→end), XeqY(Req21→start,Req10→end), XeqY(Req21→start,Req11→end),
XeqY(Req21→start,Req12→end), XeqY(Req21→start,Req13→end), XeqY(Req21→start,Req14→end), XeqY(Req21→start,Req15→end),
XeqY(Req21→start,Req16→end), XeqY(Req21→start,Req17→end), XeqY(Req21→start,Req18→end), XeqY(Req21→start,Req19→end),
XeqY(Req21→start,Req20→end), XeqY(Req21→start,Req22→end), XeqY(Req21→start,Req23→end), XeqY(Req21→start,Req24→end),
XeqY(Req21→start,Req25→end), XeqY(Req21→start,Req26→end), XeqY(Req21→start,Req27→end), XeqY(Req21→start,Req28→end),
XeqY(Req21→start,Req29→end), XeqY(Req21→start,Req30→end), XeqY(Req21→start,Req31→end), XeqY(Req21→start,Req32→end),
XeqY(Req21→start,Req33→end), XeqY(Req21→start,Req34→end), XeqY(Req21→start,Req35→end), XeqY(Req21→start,Req36→end),
XeqY(Req21→start,Req37→end), XeqY(Req21→start,Req38→end), XeqY(Req21→start,Req39→end), XeqY(Req21→start,Req40→end),
XeqY(Req21→start,Req41→end), XeqY(Req21→start,Req42→end), XeqY(Req21→start,Req43→end), XeqY(Req21→start,Req44→end),
XeqY(Req21→start,Req45→end), XeqY(Req21→start,Req46→end), XeqY(Req21→start,Req47→end), XeqY(Req21→start,Req48→end),
XeqY(Req21→start,Rel0→start), XeqY(Req21→start,Rel1→start), XeqY(Req21→start,Rel2→start), XeqY(Req21→start, project_end)))

Possible starts of Economic efficiency: Or(
XeqY(Req22→start,Req0→end), XeqY(Req22→start,Req1→end), XeqY(Req22→start,Req2→end), XeqY(Req22→start,Req3→end),
XeqY(Req22→start,Req4→end), XeqY(Req22→start,Req5→end), XeqY(Req22→start,Req6→end), XeqY(Req22→start,Req7→end),
XeqY(Req22→start,Req8→end), XeqY(Req22→start,Req9→end), XeqY(Req22→start,Req10→end), XeqY(Req22→start,Req11→end),
XeqY(Req22→start,Req12→end), XeqY(Req22→start,Req13→end), XeqY(Req22→start,Req14→end), XeqY(Req22→start,Req15→end),
XeqY(Req22→start,Req16→end), XeqY(Req22→start,Req17→end), XeqY(Req22→start,Req18→end), XeqY(Req22→start,Req19→end),
XeqY(Req22→start,Req20→end), XeqY(Req22→start,Req21→end), XeqY(Req22→start,Req23→end), XeqY(Req22→start,Req24→end),
XeqY(Req22→start,Req25→end), XeqY(Req22→start,Req26→end), XeqY(Req22→start,Req27→end), XeqY(Req22→start,Req28→end),
XeqY(Req22→start,Req29→end), XeqY(Req22→start,Req30→end), XeqY(Req22→start,Req31→end), XeqY(Req22→start,Req32→end),
XeqY(Req22→start,Req33→end), XeqY(Req22→start,Req34→end), XeqY(Req22→start,Req35→end), XeqY(Req22→start,Req36→end),
XeqY(Req22→start,Req37→end), XeqY(Req22→start,Req38→end), XeqY(Req22→start,Req39→end), XeqY(Req22→start,Req40→end),
XeqY(Req22→start,Req41→end), XeqY(Req22→start,Req42→end), XeqY(Req22→start,Req43→end), XeqY(Req22→start,Req44→end),
XeqY(Req22→start,Req45→end), XeqY(Req22→start,Req46→end), XeqY(Req22→start,Req47→end), XeqY(Req22→start,Req48→end),
XeqY(Req22→start,Rel0→start), XeqY(Req22→start,Rel1→start), XeqY(Req22→start,Rel2→start), XeqY(Req22→start, project_end)))

Possible starts of Collectors and installation (GUI): Or(
XeqY(Req23→start,Req0→end), XeqY(Req23→start,Req1→end), XeqY(Req23→start,Req2→end), XeqY(Req23→start,Req3→end),
XeqY(Req23→start,Req4→end), XeqY(Req23→start,Req5→end), XeqY(Req23→start,Req6→end), XeqY(Req23→start,Req7→end),
XeqY(Req23→start,Req8→end), XeqY(Req23→start,Req9→end), XeqY(Req23→start,Req10→end), XeqY(Req23→start,Req11→end),
XeqY(Req23→start,Req12→end), XeqY(Req23→start,Req13→end), XeqY(Req23→start,Req14→end), XeqY(Req23→start,Req15→end),
XeqY(Req23→start,Req16→end), XeqY(Req23→start,Req17→end), XeqY(Req23→start,Req18→end), XeqY(Req23→start,Req19→end),
XeqY(Req23→start,Req20→end), XeqY(Req23→start,Req21→end), XeqY(Req23→start,Req22→end), XeqY(Req23→start,Req24→end),
XeqY(Req23→start,Req25→end), XeqY(Req23→start,Req26→end), XeqY(Req23→start,Req27→end), XeqY(Req23→start,Req28→end),
XeqY(Req23→start,Req29→end), XeqY(Req23→start,Req30→end), XeqY(Req23→start,Req31→end), XeqY(Req23→start,Req32→end),
XeqY(Req23→start,Req33→end), XeqY(Req23→start,Req34→end), XeqY(Req23→start,Req35→end), XeqY(Req23→start,Req36→end),
XeqY(Req23→start,Req37→end), XeqY(Req23→start,Req38→end), XeqY(Req23→start,Req39→end), XeqY(Req23→start,Req40→end),
XeqY(Req23→start,Req41→end), XeqY(Req23→start,Req42→end), XeqY(Req23→start,Req43→end), XeqY(Req23→start,Req44→end),
XeqY(Req23→start,Req45→end), XeqY(Req23→start,Req46→end), XeqY(Req23→start,Req47→end), XeqY(Req23→start,Req48→end),
XeqY(Req23→start,Rel0→start), XeqY(Req23→start,Rel1→start), XeqY(Req23→start,Rel2→start), XeqY(Req23→start, project_end)))

Possible starts of Pipe costs (GUI-input): Or(
XeqY(Req24→start,Req0→end), XeqY(Req24→start,Req1→end), XeqY(Req24→start,Req2→end), XeqY(Req24→start,Req3→end),
XeqY(Req24→start,Req4→end), XeqY(Req24→start,Req5→end), XeqY(Req24→start,Req6→end), XeqY(Req24→start,Req7→end),
XeqY(Req24→start,Req8→end), XeqY(Req24→start,Req9→end), XeqY(Req24→start,Req10→end), XeqY(Req24→start,Req11→end),
XeqY(Req24→start,Req12→end), XeqY(Req24→start,Req13→end), XeqY(Req24→start,Req14→end), XeqY(Req24→start,Req15→end),
XeqY(Req24→start,Req16→end), XeqY(Req24→start,Req17→end), XeqY(Req24→start,Req18→end), XeqY(Req24→start,Req19→end),
XeqY(Req24→start,Req20→end), XeqY(Req24→start,Req21→end), XeqY(Req24→start,Req22→end), XeqY(Req24→start,Req23→end),
XeqY(Req24→start,Req25→end), XeqY(Req24→start,Req26→end), XeqY(Req24→start,Req27→end), XeqY(Req24→start,Req28→end),
XeqY(Req24→start,Req29→end), XeqY(Req24→start,Req30→end), XeqY(Req24→start,Req31→end), XeqY(Req24→start,Req32→end),
XeqY(Req24→start,Req33→end), XeqY(Req24→start,Req34→end), XeqY(Req24→start,Req35→end), XeqY(Req24→start,Req36→end),
XeqY(Req24→start,Req37→end), XeqY(Req24→start,Req38→end), XeqY(Req24→start,Req39→end), XeqY(Req24→start,Req40→end),
XeqY(Req24→start,Req41→end), XeqY(Req24→start,Req42→end), XeqY(Req24→start,Req43→end), XeqY(Req24→start,Req44→end),
XeqY(Req24→start,Req45→end), XeqY(Req24→start,Req46→end), XeqY(Req24→start,Req47→end), XeqY(Req24→start,Req48→end),
XeqY(Req24→start,Rel0→start), XeqY(Req24→start,Rel1→start), XeqY(Req24→start,Rel2→start), XeqY(Req24→start, project_end)))

Possible starts of Air caps calculation (GUI-Input): Or(
XeqY(Req25→start,Req0→end), XeqY(Req25→start,Req1→end), XeqY(Req25→start,Req2→end), XeqY(Req25→start,Req3→end),
XeqY(Req25→start,Req4→end), XeqY(Req25→start,Req5→end), XeqY(Req25→start,Req6→end), XeqY(Req25→start,Req7→end),
XeqY(Req25→start,Req8→end), XeqY(Req25→start,Req9→end), XeqY(Req25→start,Req10→end), XeqY(Req25→start,Req11→end),
XeqY(Req25→start,Req12→end), XeqY(Req25→start,Req13→end), XeqY(Req25→start,Req14→end), XeqY(Req25→start,Req15→end),
XeqY(Req25→start,Req16→end), XeqY(Req25→start,Req17→end), XeqY(Req25→start,Req18→end), XeqY(Req25→start,Req19→end),
XeqY(Req25→start,Req20→end), XeqY(Req25→start,Req21→end), XeqY(Req25→start,Req22→end), XeqY(Req25→start,Req23→end),
XeqY(Req25→start,Req24→end), XeqY(Req25→start,Req26→end), XeqY(Req25→start,Req27→end), XeqY(Req25→start,Req28→end),
XeqY(Req25→start,Req29→end), XeqY(Req25→start,Req30→end), XeqY(Req25→start,Req31→end), XeqY(Req25→start,Req32→end),
XeqY(Req25→start,Req33→end), XeqY(Req25→start,Req34→end), XeqY(Req25→start,Req35→end), XeqY(Req25→start,Req36→end),
XeqY(Req25→start,Req37→end), XeqY(Req25→start,Req38→end), XeqY(Req25→start,Req39→end), XeqY(Req25→start,Req40→end),
XeqY(Req25→start,Req41→end), XeqY(Req25→start,Req42→end), XeqY(Req25→start,Req43→end), XeqY(Req25→start,Req44→end),
XeqY(Req25→start,Req45→end), XeqY(Req25→start,Req46→end), XeqY(Req25→start,Req47→end), XeqY(Req25→start,Req48→end),
XeqY(Req25→start,Rel0→start), XeqY(Req25→start,Rel1→start), XeqY(Req25→start,Rel2→start), XeqY(Req25→start, project_end)))

Possible starts of PGR & ADG (GUI-Input): Or(
XeqY(Req26→start,Req0→end), XeqY(Req26→start,Req1→end), XeqY(Req26→start,Req2→end), XeqY(Req26→start,Req3→end),
XeqY(Req26→start,Req4→end), XeqY(Req26→start,Req5→end), XeqY(Req26→start,Req6→end), XeqY(Req26→start,Req7→end),
XeqY(Req26→start,Req8→end), XeqY(Req26→start,Req9→end), XeqY(Req26→start,Req10→end), XeqY(Req26→start,Req11→end),
XeqY(Req26→start,Req12→end), XeqY(Req26→start,Req13→end), XeqY(Req26→start,Req14→end), XeqY(Req26→start,Req15→end),
XeqY(Req26→start,Req16→end), XeqY(Req26→start,Req17→end), XeqY(Req26→start,Req18→end), XeqY(Req26→start,Req19→end),
XeqY(Req26→start,Req20→end), XeqY(Req26→start,Req21→end), XeqY(Req26→start,Req22→end), XeqY(Req26→start,Req23→end),
XeqY(Req26→start,Req24→end), XeqY(Req26→start,Req25→end), XeqY(Req26→start,Req27→end), XeqY(Req26→start,Req28→end),
XeqY(Req26→start,Req29→end), XeqY(Req26→start,Req30→end), XeqY(Req26→start,Req31→end), XeqY(Req26→start,Req32→end),
XeqY(Req26→start,Req33→end), XeqY(Req26→start,Req34→end), XeqY(Req26→start,Req35→end), XeqY(Req26→start,Req36→end),
XeqY(Req26→start,Req37→end), XeqY(Req26→start,Req38→end), XeqY(Req26→start,Req39→end), XeqY(Req26→start,Req40→end),
XeqY(Req26→start,Req41→end), XeqY(Req26→start,Req42→end), XeqY(Req26→start,Req43→end), XeqY(Req26→start,Req44→end),
XeqY(Req26→start,Req45→end), XeqY(Req26→start,Req46→end), XeqY(Req26→start,Req47→end), XeqY(Req26→start,Req48→end),
XeqY(Req26→start,Rel0→start), XeqY(Req26→start,Rel1→start), XeqY(Req26→start,Rel2→start), XeqY(Req26→start, project_end)))

Possible starts of Costs for refilling (Calculation): Or(
XeqY(Req27→start,Req0→end), XeqY(Req27→start,Req1→end), XeqY(Req27→start,Req2→end), XeqY(Req27→start,Req3→end),
XeqY(Req27→start,Req4→end), XeqY(Req27→start,Req5→end), XeqY(Req27→start,Req6→end), XeqY(Req27→start,Req7→end),
XeqY(Req27→start,Req8→end), XeqY(Req27→start,Req9→end), XeqY(Req27→start,Req10→end), XeqY(Req27→start,Req11→end),
XeqY(Req27→start,Req12→end), XeqY(Req27→start,Req13→end), XeqY(Req27→start,Req14→end), XeqY(Req27→start,Req15→end),
XeqY(Req27→start,Req16→end), XeqY(Req27→start,Req17→end), XeqY(Req27→start,Req18→end), XeqY(Req27→start,Req19→end),
XeqY(Req27→start,Req20→end), XeqY(Req27→start,Req21→end), XeqY(Req27→start,Req22→end), XeqY(Req27→start,Req23→end),
XeqY(Req27→start,Req24→end), XeqY(Req27→start,Req25→end), XeqY(Req27→start,Req26→end), XeqY(Req27→start,Req28→end),
XeqY(Req27→start,Req29→end), XeqY(Req27→start,Req30→end), XeqY(Req27→start,Req31→end), XeqY(Req27→start,Req32→end),
XeqY(Req27→start,Req33→end), XeqY(Req27→start,Req34→end), XeqY(Req27→start,Req35→end), XeqY(Req27→start,Req36→end),
XeqY(Req27→start,Req37→end), XeqY(Req27→start,Req38→end), XeqY(Req27→start,Req39→end), XeqY(Req27→start,Req40→end),
XeqY(Req27→start,Req41→end), XeqY(Req27→start,Req42→end), XeqY(Req27→start,Req43→end), XeqY(Req27→start,Req44→end),
XeqY(Req27→start,Req45→end), XeqY(Req27→start,Req46→end), XeqY(Req27→start,Req47→end), XeqY(Req27→start,Req48→end),
XeqY(Req27→start,Rel0→start), XeqY(Req27→start,Rel1→start), XeqY(Req27→start,Rel2→start), XeqY(Req27→start, project_end)))

Possible starts of Special Equipment (GUI-input): Or(
XeqY(Req28→start,Req0→end), XeqY(Req28→start,Req1→end), XeqY(Req28→start,Req2→end), XeqY(Req28→start,Req3→end),
XeqY(Req28→start,Req4→end), XeqY(Req28→start,Req5→end), XeqY(Req28→start,Req6→end), XeqY(Req28→start,Req7→end),
XeqY(Req28→start,Req8→end), XeqY(Req28→start,Req9→end), XeqY(Req28→start,Req10→end), XeqY(Req28→start,Req11→end),
XeqY(Req28→start,Req12→end), XeqY(Req28→start,Req13→end), XeqY(Req28→start,Req14→end), XeqY(Req28→start,Req15→end),
XeqY(Req28→start,Req16→end), XeqY(Req28→start,Req17→end), XeqY(Req28→start,Req18→end), XeqY(Req28→start,Req19→end),
XeqY(Req28→start,Req20→end), XeqY(Req28→start,Req21→end), XeqY(Req28→start,Req22→end), XeqY(Req28→start,Req23→end),
XeqY(Req28→start,Req24→end), XeqY(Req28→start,Req25→end), XeqY(Req28→start,Req26→end), XeqY(Req28→start,Req27→end),
XeqY(Req28→start,Req29→end), XeqY(Req28→start,Req30→end), XeqY(Req28→start,Req31→end), XeqY(Req28→start,Req32→end),
XeqY(Req28→start,Req33→end), XeqY(Req28→start,Req34→end), XeqY(Req28→start,Req35→end), XeqY(Req28→start,Req36→end),
XeqY(Req28→start,Req37→end), XeqY(Req28→start,Req38→end), XeqY(Req28→start,Req39→end), XeqY(Req28→start,Req40→end),
XeqY(Req28→start,Req41→end), XeqY(Req28→start,Req42→end), XeqY(Req28→start,Req43→end), XeqY(Req28→start,Req44→end),
XeqY(Req28→start,Req45→end), XeqY(Req28→start,Req46→end), XeqY(Req28→start,Req47→end), XeqY(Req28→start,Req48→end),
XeqY(Req28→start,Rel0→start), XeqY(Req28→start,Rel1→start), XeqY(Req28→start,Rel2→start), XeqY(Req28→start, project_end)))

Possible starts of PGR & ADG (Calculation) : Or(
XeqY(Req29→start,Req0→end), XeqY(Req29→start,Req1→end), XeqY(Req29→start,Req2→end), XeqY(Req29→start,Req3→end),
XeqY(Req29→start,Req4→end), XeqY(Req29→start,Req5→end), XeqY(Req29→start,Req6→end), XeqY(Req29→start,Req7→end),
XeqY(Req29→start,Req8→end), XeqY(Req29→start,Req9→end), XeqY(Req29→start,Req10→end), XeqY(Req29→start,Req11→end),
XeqY(Req29→start,Req12→end), XeqY(Req29→start,Req13→end), XeqY(Req29→start,Req14→end), XeqY(Req29→start,Req15→end),
XeqY(Req29→start,Req16→end), XeqY(Req29→start,Req17→end), XeqY(Req29→start,Req18→end), XeqY(Req29→start,Req19→end),
XeqY(Req29→start,Req20→end), XeqY(Req29→start,Req21→end), XeqY(Req29→start,Req22→end), XeqY(Req29→start,Req23→end),
XeqY(Req29→start,Req24→end), XeqY(Req29→start,Req25→end), XeqY(Req29→start,Req26→end), XeqY(Req29→start,Req27→end),
XeqY(Req29→start,Req28→end), XeqY(Req29→start,Req30→end), XeqY(Req29→start,Req31→end), XeqY(Req29→start,Req32→end),
XeqY(Req29→start,Req33→end), XeqY(Req29→start,Req34→end), XeqY(Req29→start,Req35→end), XeqY(Req29→start,Req36→end),
XeqY(Req29→start,Req37→end), XeqY(Req29→start,Req38→end), XeqY(Req29→start,Req39→end), XeqY(Req29→start,Req40→end),
XeqY(Req29→start,Req41→end), XeqY(Req29→start,Req42→end), XeqY(Req29→start,Req43→end), XeqY(Req29→start,Req44→end),
XeqY(Req29→start,Req45→end), XeqY(Req29→start,Req46→end), XeqY(Req29→start,Req47→end), XeqY(Req29→start,Req48→end),
XeqY(Req29→start,Rel0→start), XeqY(Req29→start,Rel1→start), XeqY(Req29→start,Rel2→start), XeqY(Req29→start, project_end)))

Possible starts of Eco. efficiency (Calc.): Or(
XeqY(Req30→start,Req0→end), XeqY(Req30→start,Req1→end), XeqY(Req30→start,Req2→end), XeqY(Req30→start,Req3→end),
XeqY(Req30→start,Req4→end), XeqY(Req30→start,Req5→end), XeqY(Req30→start,Req6→end), XeqY(Req30→start,Req7→end),
XeqY(Req30→start,Req8→end), XeqY(Req30→start,Req9→end), XeqY(Req30→start,Req10→end), XeqY(Req30→start,Req11→end),
XeqY(Req30→start,Req12→end), XeqY(Req30→start,Req13→end), XeqY(Req30→start,Req14→end), XeqY(Req30→start,Req15→end),
XeqY(Req30→start,Req16→end), XeqY(Req30→start,Req17→end), XeqY(Req30→start,Req18→end), XeqY(Req30→start,Req19→end),
XeqY(Req30→start,Req20→end), XeqY(Req30→start,Req21→end), XeqY(Req30→start,Req22→end), XeqY(Req30→start,Req23→end),
XeqY(Req30→start,Req24→end), XeqY(Req30→start,Req25→end), XeqY(Req30→start,Req26→end), XeqY(Req30→start,Req27→end),
XeqY(Req30→start,Req28→end), XeqY(Req30→start,Req29→end), XeqY(Req30→start,Req31→end), XeqY(Req30→start,Req32→end),
XeqY(Req30→start,Req33→end), XeqY(Req30→start,Req34→end), XeqY(Req30→start,Req35→end), XeqY(Req30→start,Req36→end),
XeqY(Req30→start,Req37→end), XeqY(Req30→start,Req38→end), XeqY(Req30→start,Req39→end), XeqY(Req30→start,Req40→end),
XeqY(Req30→start,Req41→end), XeqY(Req30→start,Req42→end), XeqY(Req30→start,Req43→end), XeqY(Req30→start,Req44→end),
XeqY(Req30→start,Req45→end), XeqY(Req30→start,Req46→end), XeqY(Req30→start,Req47→end), XeqY(Req30→start,Req48→end),
XeqY(Req30→start,Rel0→start), XeqY(Req30→start,Rel1→start), XeqY(Req30→start,Rel2→start), XeqY(Req30→start, project_end)))

Possible starts of Eco. efficiency (GUI-input): Or(
XeqY(Req31→start,Req0→end), XeqY(Req31→start,Req1→end), XeqY(Req31→start,Req2→end), XeqY(Req31→start,Req3→end),
XeqY(Req31→start,Req4→end), XeqY(Req31→start,Req5→end), XeqY(Req31→start,Req6→end), XeqY(Req31→start,Req7→end),
XeqY(Req31→start,Req8→end), XeqY(Req31→start,Req9→end), XeqY(Req31→start,Req10→end), XeqY(Req31→start,Req11→end),
XeqY(Req31→start,Req12→end), XeqY(Req31→start,Req13→end), XeqY(Req31→start,Req14→end), XeqY(Req31→start,Req15→end),
XeqY(Req31→start,Req16→end), XeqY(Req31→start,Req17→end), XeqY(Req31→start,Req18→end), XeqY(Req31→start,Req19→end),
XeqY(Req31→start,Req20→end), XeqY(Req31→start,Req21→end), XeqY(Req31→start,Req22→end), XeqY(Req31→start,Req23→end),
XeqY(Req31→start,Req24→end), XeqY(Req31→start,Req25→end), XeqY(Req31→start,Req26→end), XeqY(Req31→start,Req27→end),
XeqY(Req31→start,Req28→end), XeqY(Req31→start,Req29→end), XeqY(Req31→start,Req30→end), XeqY(Req31→start,Req32→end),
XeqY(Req31→start,Req33→end), XeqY(Req31→start,Req34→end), XeqY(Req31→start,Req35→end), XeqY(Req31→start,Req36→end),
XeqY(Req31→start,Req37→end), XeqY(Req31→start,Req38→end), XeqY(Req31→start,Req39→end), XeqY(Req31→start,Req40→end),
XeqY(Req31→start,Req41→end), XeqY(Req31→start,Req42→end), XeqY(Req31→start,Req43→end), XeqY(Req31→start,Req44→end),
XeqY(Req31→start,Req45→end), XeqY(Req31→start,Req46→end), XeqY(Req31→start,Req47→end), XeqY(Req31→start,Req48→end),
XeqY(Req31→start,Rel0→start), XeqY(Req31→start,Rel1→start), XeqY(Req31→start,Rel2→start), XeqY(Req31→start, project_end)))

Possible starts of Eco. efficiency (Proposal): Or(
XeqY(Req32→start,Req0→end), XeqY(Req32→start,Req1→end), XeqY(Req32→start,Req2→end), XeqY(Req32→start,Req3→end),
XeqY(Req32→start,Req4→end), XeqY(Req32→start,Req5→end), XeqY(Req32→start,Req6→end), XeqY(Req32→start,Req7→end),
XeqY(Req32→start,Req8→end), XeqY(Req32→start,Req9→end), XeqY(Req32→start,Req10→end), XeqY(Req32→start,Req11→end),
XeqY(Req32→start,Req12→end), XeqY(Req32→start,Req13→end), XeqY(Req32→start,Req14→end), XeqY(Req32→start,Req15→end),
XeqY(Req32→start,Req16→end), XeqY(Req32→start,Req17→end), XeqY(Req32→start,Req18→end), XeqY(Req32→start,Req19→end),
XeqY(Req32→start,Req20→end), XeqY(Req32→start,Req21→end), XeqY(Req32→start,Req22→end), XeqY(Req32→start,Req23→end),
XeqY(Req32→start,Req24→end), XeqY(Req32→start,Req25→end), XeqY(Req32→start,Req26→end), XeqY(Req32→start,Req27→end),
XeqY(Req32→start,Req28→end), XeqY(Req32→start,Req29→end), XeqY(Req32→start,Req30→end), XeqY(Req32→start,Req31→end),
XeqY(Req32→start,Req33→end), XeqY(Req32→start,Req34→end), XeqY(Req32→start,Req35→end), XeqY(Req32→start,Req36→end),
XeqY(Req32→start,Req37→end), XeqY(Req32→start,Req38→end), XeqY(Req32→start,Req39→end), XeqY(Req32→start,Req40→end),
XeqY(Req32→start,Req41→end), XeqY(Req32→start,Req42→end), XeqY(Req32→start,Req43→end), XeqY(Req32→start,Req44→end),
XeqY(Req32→start,Req45→end), XeqY(Req32→start,Req46→end), XeqY(Req32→start,Req47→end), XeqY(Req32→start,Req48→end),
XeqY(Req32→start,Rel0→start), XeqY(Req32→start,Rel1→start), XeqY(Req32→start,Rel2→start), XeqY(Req32→start, project_end)))

Possible starts of Modelling an Iconset: Or(
XeqY(Req33→start,Req0→end), XeqY(Req33→start,Req1→end), XeqY(Req33→start,Req2→end), XeqY(Req33→start,Req3→end),
XeqY(Req33→start,Req4→end), XeqY(Req33→start,Req5→end), XeqY(Req33→start,Req6→end), XeqY(Req33→start,Req7→end),
XeqY(Req33→start,Req8→end), XeqY(Req33→start,Req9→end), XeqY(Req33→start,Req10→end), XeqY(Req33→start,Req11→end),
XeqY(Req33→start,Req12→end), XeqY(Req33→start,Req13→end), XeqY(Req33→start,Req14→end), XeqY(Req33→start,Req15→end),
XeqY(Req33→start,Req16→end), XeqY(Req33→start,Req17→end), XeqY(Req33→start,Req18→end), XeqY(Req33→start,Req19→end),
XeqY(Req33→start,Req20→end), XeqY(Req33→start,Req21→end), XeqY(Req33→start,Req22→end), XeqY(Req33→start,Req23→end),
XeqY(Req33→start,Req24→end), XeqY(Req33→start,Req25→end), XeqY(Req33→start,Req26→end), XeqY(Req33→start,Req27→end),
XeqY(Req33→start,Req28→end), XeqY(Req33→start,Req29→end), XeqY(Req33→start,Req30→end), XeqY(Req33→start,Req31→end),
XeqY(Req33→start,Req32→end), XeqY(Req33→start,Req34→end), XeqY(Req33→start,Req35→end), XeqY(Req33→start,Req36→end),
XeqY(Req33→start,Req37→end), XeqY(Req33→start,Req38→end), XeqY(Req33→start,Req39→end), XeqY(Req33→start,Req40→end),
XeqY(Req33→start,Req41→end), XeqY(Req33→start,Req42→end), XeqY(Req33→start,Req43→end), XeqY(Req33→start,Req44→end),
XeqY(Req33→start,Req45→end), XeqY(Req33→start,Req46→end), XeqY(Req33→start,Req47→end), XeqY(Req33→start,Req48→end),
XeqY(Req33→start,Rel0→start), XeqY(Req33→start,Rel1→start), XeqY(Req33→start,Rel2→start), XeqY(Req33→start, project_end)))

Possible starts of Mysql Backend: Or(
XeqY(Req34→start,Req0→end), XeqY(Req34→start,Req1→end), XeqY(Req34→start,Req2→end), XeqY(Req34→start,Req3→end),
XeqY(Req34→start,Req4→end), XeqY(Req34→start,Req5→end), XeqY(Req34→start,Req6→end), XeqY(Req34→start,Req7→end),
XeqY(Req34→start,Req8→end), XeqY(Req34→start,Req9→end), XeqY(Req34→start,Req10→end), XeqY(Req34→start,Req11→end),
XeqY(Req34→start,Req12→end), XeqY(Req34→start,Req13→end), XeqY(Req34→start,Req14→end), XeqY(Req34→start,Req15→end),
XeqY(Req34→start,Req16→end), XeqY(Req34→start,Req17→end), XeqY(Req34→start,Req18→end), XeqY(Req34→start,Req19→end),
XeqY(Req34→start,Req20→end), XeqY(Req34→start,Req21→end), XeqY(Req34→start,Req22→end), XeqY(Req34→start,Req23→end),
XeqY(Req34→start,Req24→end), XeqY(Req34→start,Req25→end), XeqY(Req34→start,Req26→end), XeqY(Req34→start,Req27→end),
XeqY(Req34→start,Req28→end), XeqY(Req34→start,Req29→end), XeqY(Req34→start,Req30→end), XeqY(Req34→start,Req31→end),
XeqY(Req34→start,Req32→end), XeqY(Req34→start,Req33→end), XeqY(Req34→start,Req35→end), XeqY(Req34→start,Req36→end),
XeqY(Req34→start,Req37→end), XeqY(Req34→start,Req38→end), XeqY(Req34→start,Req39→end), XeqY(Req34→start,Req40→end),
XeqY(Req34→start,Req41→end), XeqY(Req34→start,Req42→end), XeqY(Req34→start,Req43→end), XeqY(Req34→start,Req44→end),
XeqY(Req34→start,Req45→end), XeqY(Req34→start,Req46→end), XeqY(Req34→start,Req47→end), XeqY(Req34→start,Req48→end),
XeqY(Req34→start,Rel0→start), XeqY(Req34→start,Rel1→start), XeqY(Req34→start,Rel2→start), XeqY(Req34→start, project_end)))

Possible starts of Sqlite Backend: Or(
XeqY(Req35→start,Req0→end), XeqY(Req35→start,Req1→end), XeqY(Req35→start,Req2→end), XeqY(Req35→start,Req3→end),
XeqY(Req35→start,Req4→end), XeqY(Req35→start,Req5→end), XeqY(Req35→start,Req6→end), XeqY(Req35→start,Req7→end),
XeqY(Req35→start,Req8→end), XeqY(Req35→start,Req9→end), XeqY(Req35→start,Req10→end), XeqY(Req35→start,Req11→end),
XeqY(Req35→start,Req12→end), XeqY(Req35→start,Req13→end), XeqY(Req35→start,Req14→end), XeqY(Req35→start,Req15→end),
XeqY(Req35→start,Req16→end), XeqY(Req35→start,Req17→end), XeqY(Req35→start,Req18→end), XeqY(Req35→start,Req19→end),
XeqY(Req35→start,Req20→end), XeqY(Req35→start,Req21→end), XeqY(Req35→start,Req22→end), XeqY(Req35→start,Req23→end),
XeqY(Req35→start,Req24→end), XeqY(Req35→start,Req25→end), XeqY(Req35→start,Req26→end), XeqY(Req35→start,Req27→end),
XeqY(Req35→start,Req28→end), XeqY(Req35→start,Req29→end), XeqY(Req35→start,Req30→end), XeqY(Req35→start,Req31→end),
XeqY(Req35→start,Req32→end), XeqY(Req35→start,Req33→end), XeqY(Req35→start,Req34→end), XeqY(Req35→start,Req36→end),
XeqY(Req35→start,Req37→end), XeqY(Req35→start,Req38→end), XeqY(Req35→start,Req39→end), XeqY(Req35→start,Req40→end),
XeqY(Req35→start,Req41→end), XeqY(Req35→start,Req42→end), XeqY(Req35→start,Req43→end), XeqY(Req35→start,Req44→end),
XeqY(Req35→start,Req45→end), XeqY(Req35→start,Req46→end), XeqY(Req35→start,Req47→end), XeqY(Req35→start,Req48→end),
XeqY(Req35→start,Rel0→start), XeqY(Req35→start,Rel1→start), XeqY(Req35→start,Rel2→start), XeqY(Req35→start, project_end)))

Possible starts of Costs for refilling (GUI-input): Or(
XeqY(Req36→start,Req0→end), XeqY(Req36→start,Req1→end), XeqY(Req36→start,Req2→end), XeqY(Req36→start,Req3→end),
XeqY(Req36→start,Req4→end), XeqY(Req36→start,Req5→end), XeqY(Req36→start,Req6→end), XeqY(Req36→start,Req7→end),
XeqY(Req36→start,Req8→end), XeqY(Req36→start,Req9→end), XeqY(Req36→start,Req10→end), XeqY(Req36→start,Req11→end),
XeqY(Req36→start,Req12→end), XeqY(Req36→start,Req13→end), XeqY(Req36→start,Req14→end), XeqY(Req36→start,Req15→end),
XeqY(Req36→start,Req16→end), XeqY(Req36→start,Req17→end), XeqY(Req36→start,Req18→end), XeqY(Req36→start,Req19→end),
XeqY(Req36→start,Req20→end), XeqY(Req36→start,Req21→end), XeqY(Req36→start,Req22→end), XeqY(Req36→start,Req23→end),
XeqY(Req36→start,Req24→end), XeqY(Req36→start,Req25→end), XeqY(Req36→start,Req26→end), XeqY(Req36→start,Req27→end),
XeqY(Req36→start,Req28→end), XeqY(Req36→start,Req29→end), XeqY(Req36→start,Req30→end), XeqY(Req36→start,Req31→end),
XeqY(Req36→start,Req32→end), XeqY(Req36→start,Req33→end), XeqY(Req36→start,Req34→end), XeqY(Req36→start,Req35→end),
XeqY(Req36→start,Req37→end), XeqY(Req36→start,Req38→end), XeqY(Req36→start,Req39→end), XeqY(Req36→start,Req40→end),
XeqY(Req36→start,Req41→end), XeqY(Req36→start,Req42→end), XeqY(Req36→start,Req43→end), XeqY(Req36→start,Req44→end),
XeqY(Req36→start,Req45→end), XeqY(Req36→start,Req46→end), XeqY(Req36→start,Req47→end), XeqY(Req36→start,Req48→end),
XeqY(Req36→start,Rel0→start), XeqY(Req36→start,Rel1→start), XeqY(Req36→start,Rel2→start), XeqY(Req36→start, project_end)))

Possible starts of Apache Authentication: Or(
XeqY(Req37→start,Req0→end), XeqY(Req37→start,Req1→end), XeqY(Req37→start,Req2→end), XeqY(Req37→start,Req3→end),
XeqY(Req37→start,Req4→end), XeqY(Req37→start,Req5→end), XeqY(Req37→start,Req6→end), XeqY(Req37→start,Req7→end),
XeqY(Req37→start,Req8→end), XeqY(Req37→start,Req9→end), XeqY(Req37→start,Req10→end), XeqY(Req37→start,Req11→end),
XeqY(Req37→start,Req12→end), XeqY(Req37→start,Req13→end), XeqY(Req37→start,Req14→end), XeqY(Req37→start,Req15→end),
XeqY(Req37→start,Req16→end), XeqY(Req37→start,Req17→end), XeqY(Req37→start,Req18→end), XeqY(Req37→start,Req19→end),
XeqY(Req37→start,Req20→end), XeqY(Req37→start,Req21→end), XeqY(Req37→start,Req22→end), XeqY(Req37→start,Req23→end),
XeqY(Req37→start,Req24→end), XeqY(Req37→start,Req25→end), XeqY(Req37→start,Req26→end), XeqY(Req37→start,Req27→end),
XeqY(Req37→start,Req28→end), XeqY(Req37→start,Req29→end), XeqY(Req37→start,Req30→end), XeqY(Req37→start,Req31→end),
XeqY(Req37→start,Req32→end), XeqY(Req37→start,Req33→end), XeqY(Req37→start,Req34→end), XeqY(Req37→start,Req35→end),
XeqY(Req37→start,Req36→end), XeqY(Req37→start,Req38→end), XeqY(Req37→start,Req39→end), XeqY(Req37→start,Req40→end),
XeqY(Req37→start,Req41→end), XeqY(Req37→start,Req42→end), XeqY(Req37→start,Req43→end), XeqY(Req37→start,Req44→end),
XeqY(Req37→start,Req45→end), XeqY(Req37→start,Req46→end), XeqY(Req37→start,Req47→end), XeqY(Req37→start,Req48→end),
XeqY(Req37→start,Rel0→start), XeqY(Req37→start,Rel1→start), XeqY(Req37→start,Rel2→start), XeqY(Req37→start, project_end)))

Possible starts of DB Auth.: Or(
XeqY(Req38→start,Req0→end), XeqY(Req38→start,Req1→end), XeqY(Req38→start,Req2→end), XeqY(Req38→start,Req3→end),
XeqY(Req38→start,Req4→end), XeqY(Req38→start,Req5→end), XeqY(Req38→start,Req6→end), XeqY(Req38→start,Req7→end),
XeqY(Req38→start,Req8→end), XeqY(Req38→start,Req9→end), XeqY(Req38→start,Req10→end), XeqY(Req38→start,Req11→end),
XeqY(Req38→start,Req12→end), XeqY(Req38→start,Req13→end), XeqY(Req38→start,Req14→end), XeqY(Req38→start,Req15→end),
XeqY(Req38→start,Req16→end), XeqY(Req38→start,Req17→end), XeqY(Req38→start,Req18→end), XeqY(Req38→start,Req19→end),
XeqY(Req38→start,Req20→end), XeqY(Req38→start,Req21→end), XeqY(Req38→start,Req22→end), XeqY(Req38→start,Req23→end),
XeqY(Req38→start,Req24→end), XeqY(Req38→start,Req25→end), XeqY(Req38→start,Req26→end), XeqY(Req38→start,Req27→end),
XeqY(Req38→start,Req28→end), XeqY(Req38→start,Req29→end), XeqY(Req38→start,Req30→end), XeqY(Req38→start,Req31→end),
XeqY(Req38→start,Req32→end), XeqY(Req38→start,Req33→end), XeqY(Req38→start,Req34→end), XeqY(Req38→start,Req35→end),
XeqY(Req38→start,Req36→end), XeqY(Req38→start,Req37→end), XeqY(Req38→start,Req39→end), XeqY(Req38→start,Req40→end),
XeqY(Req38→start,Req41→end), XeqY(Req38→start,Req42→end), XeqY(Req38→start,Req43→end), XeqY(Req38→start,Req44→end),
XeqY(Req38→start,Req45→end), XeqY(Req38→start,Req46→end), XeqY(Req38→start,Req47→end), XeqY(Req38→start,Req48→end),
XeqY(Req38→start,Rel0→start), XeqY(Req38→start,Rel1→start), XeqY(Req38→start,Rel2→start), XeqY(Req38→start, project_end)))

Possible starts of Auth. Tables: Or(
XeqY(Req39→start,Req0→end), XeqY(Req39→start,Req1→end), XeqY(Req39→start,Req2→end), XeqY(Req39→start,Req3→end),
XeqY(Req39→start,Req4→end), XeqY(Req39→start,Req5→end), XeqY(Req39→start,Req6→end), XeqY(Req39→start,Req7→end),
XeqY(Req39→start,Req8→end), XeqY(Req39→start,Req9→end), XeqY(Req39→start,Req10→end), XeqY(Req39→start,Req11→end),
XeqY(Req39→start,Req12→end), XeqY(Req39→start,Req13→end), XeqY(Req39→start,Req14→end), XeqY(Req39→start,Req15→end),
XeqY(Req39→start,Req16→end), XeqY(Req39→start,Req17→end), XeqY(Req39→start,Req18→end), XeqY(Req39→start,Req19→end),
XeqY(Req39→start,Req20→end), XeqY(Req39→start,Req21→end), XeqY(Req39→start,Req22→end), XeqY(Req39→start,Req23→end),
XeqY(Req39→start,Req24→end), XeqY(Req39→start,Req25→end), XeqY(Req39→start,Req26→end), XeqY(Req39→start,Req27→end),
XeqY(Req39→start,Req28→end), XeqY(Req39→start,Req29→end), XeqY(Req39→start,Req30→end), XeqY(Req39→start,Req31→end),
XeqY(Req39→start,Req32→end), XeqY(Req39→start,Req33→end), XeqY(Req39→start,Req34→end), XeqY(Req39→start,Req35→end),
XeqY(Req39→start,Req36→end), XeqY(Req39→start,Req37→end), XeqY(Req39→start,Req38→end), XeqY(Req39→start,Req40→end),
XeqY(Req39→start,Req41→end), XeqY(Req39→start,Req42→end), XeqY(Req39→start,Req43→end), XeqY(Req39→start,Req44→end),
XeqY(Req39→start,Req45→end), XeqY(Req39→start,Req46→end), XeqY(Req39→start,Req47→end), XeqY(Req39→start,Req48→end),
XeqY(Req39→start,Rel0→start), XeqY(Req39→start,Rel1→start), XeqY(Req39→start,Rel2→start), XeqY(Req39→start, project_end)))

Possible starts of Auth. DB Communication: Or(
XeqY(Req40→start,Req0→end), XeqY(Req40→start,Req1→end), XeqY(Req40→start,Req2→end), XeqY(Req40→start,Req3→end),
XeqY(Req40→start,Req4→end), XeqY(Req40→start,Req5→end), XeqY(Req40→start,Req6→end), XeqY(Req40→start,Req7→end),
XeqY(Req40→start,Req8→end), XeqY(Req40→start,Req9→end), XeqY(Req40→start,Req10→end), XeqY(Req40→start,Req11→end),
XeqY(Req40→start,Req12→end), XeqY(Req40→start,Req13→end), XeqY(Req40→start,Req14→end), XeqY(Req40→start,Req15→end),
XeqY(Req40→start,Req16→end), XeqY(Req40→start,Req17→end), XeqY(Req40→start,Req18→end), XeqY(Req40→start,Req19→end),
XeqY(Req40→start,Req20→end), XeqY(Req40→start,Req21→end), XeqY(Req40→start,Req22→end), XeqY(Req40→start,Req23→end),
XeqY(Req40→start,Req24→end), XeqY(Req40→start,Req25→end), XeqY(Req40→start,Req26→end), XeqY(Req40→start,Req27→end),
XeqY(Req40→start,Req28→end), XeqY(Req40→start,Req29→end), XeqY(Req40→start,Req30→end), XeqY(Req40→start,Req31→end),
XeqY(Req40→start,Req32→end), XeqY(Req40→start,Req33→end), XeqY(Req40→start,Req34→end), XeqY(Req40→start,Req35→end),
XeqY(Req40→start,Req36→end), XeqY(Req40→start,Req37→end), XeqY(Req40→start,Req38→end), XeqY(Req40→start,Req39→end),
XeqY(Req40→start,Req41→end), XeqY(Req40→start,Req42→end), XeqY(Req40→start,Req43→end), XeqY(Req40→start,Req44→end),
XeqY(Req40→start,Req45→end), XeqY(Req40→start,Req46→end), XeqY(Req40→start,Req47→end), XeqY(Req40→start,Req48→end),
XeqY(Req40→start,Rel0→start), XeqY(Req40→start,Rel1→start), XeqY(Req40→start,Rel2→start), XeqY(Req40→start, project_end)))

Possible starts of Auth. Pwd Encryption: Or(
XeqY(Req41→start,Req0→end), XeqY(Req41→start,Req1→end), XeqY(Req41→start,Req2→end), XeqY(Req41→start,Req3→end),
XeqY(Req41→start,Req4→end), XeqY(Req41→start,Req5→end), XeqY(Req41→start,Req6→end), XeqY(Req41→start,Req7→end),
XeqY(Req41→start,Req8→end), XeqY(Req41→start,Req9→end), XeqY(Req41→start,Req10→end), XeqY(Req41→start,Req11→end),
XeqY(Req41→start,Req12→end), XeqY(Req41→start,Req13→end), XeqY(Req41→start,Req14→end), XeqY(Req41→start,Req15→end),
XeqY(Req41→start,Req16→end), XeqY(Req41→start,Req17→end), XeqY(Req41→start,Req18→end), XeqY(Req41→start,Req19→end),
XeqY(Req41→start,Req20→end), XeqY(Req41→start,Req21→end), XeqY(Req41→start,Req22→end), XeqY(Req41→start,Req23→end),
XeqY(Req41→start,Req24→end), XeqY(Req41→start,Req25→end), XeqY(Req41→start,Req26→end), XeqY(Req41→start,Req27→end),
XeqY(Req41→start,Req28→end), XeqY(Req41→start,Req29→end), XeqY(Req41→start,Req30→end), XeqY(Req41→start,Req31→end),
XeqY(Req41→start,Req32→end), XeqY(Req41→start,Req33→end), XeqY(Req41→start,Req34→end), XeqY(Req41→start,Req35→end),
XeqY(Req41→start,Req36→end), XeqY(Req41→start,Req37→end), XeqY(Req41→start,Req38→end), XeqY(Req41→start,Req39→end),
XeqY(Req41→start,Req40→end), XeqY(Req41→start,Req42→end), XeqY(Req41→start,Req43→end), XeqY(Req41→start,Req44→end),
XeqY(Req41→start,Req45→end), XeqY(Req41→start,Req46→end), XeqY(Req41→start,Req47→end), XeqY(Req41→start,Req48→end),
XeqY(Req41→start,Rel0→start), XeqY(Req41→start,Rel1→start), XeqY(Req41→start,Rel2→start), XeqY(Req41→start, project_end)))

Possible starts of Air caps calculation (GUI-Results): Or(
XeqY(Req42→start,Req0→end), XeqY(Req42→start,Req1→end), XeqY(Req42→start,Req2→end), XeqY(Req42→start,Req3→end),
XeqY(Req42→start,Req4→end), XeqY(Req42→start,Req5→end), XeqY(Req42→start,Req6→end), XeqY(Req42→start,Req7→end),
XeqY(Req42→start,Req8→end), XeqY(Req42→start,Req9→end), XeqY(Req42→start,Req10→end), XeqY(Req42→start,Req11→end),
XeqY(Req42→start,Req12→end), XeqY(Req42→start,Req13→end), XeqY(Req42→start,Req14→end), XeqY(Req42→start,Req15→end),
XeqY(Req42→start,Req16→end), XeqY(Req42→start,Req17→end), XeqY(Req42→start,Req18→end), XeqY(Req42→start,Req19→end),
XeqY(Req42→start,Req20→end), XeqY(Req42→start,Req21→end), XeqY(Req42→start,Req22→end), XeqY(Req42→start,Req23→end),
XeqY(Req42→start,Req24→end), XeqY(Req42→start,Req25→end), XeqY(Req42→start,Req26→end), XeqY(Req42→start,Req27→end),
XeqY(Req42→start,Req28→end), XeqY(Req42→start,Req29→end), XeqY(Req42→start,Req30→end), XeqY(Req42→start,Req31→end),
XeqY(Req42→start,Req32→end), XeqY(Req42→start,Req33→end), XeqY(Req42→start,Req34→end), XeqY(Req42→start,Req35→end),
XeqY(Req42→start,Req36→end), XeqY(Req42→start,Req37→end), XeqY(Req42→start,Req38→end), XeqY(Req42→start,Req39→end),
XeqY(Req42→start,Req40→end), XeqY(Req42→start,Req41→end), XeqY(Req42→start,Req43→end), XeqY(Req42→start,Req44→end),
XeqY(Req42→start,Req45→end), XeqY(Req42→start,Req46→end), XeqY(Req42→start,Req47→end), XeqY(Req42→start,Req48→end),
XeqY(Req42→start,Rel0→start), XeqY(Req42→start,Rel1→start), XeqY(Req42→start,Rel2→start), XeqY(Req42→start, project_end)))

Possible starts of Pipe costs (GUI-results): Or(
XeqY(Req43→start,Req0→end), XeqY(Req43→start,Req1→end), XeqY(Req43→start,Req2→end), XeqY(Req43→start,Req3→end),
XeqY(Req43→start,Req4→end), XeqY(Req43→start,Req5→end), XeqY(Req43→start,Req6→end), XeqY(Req43→start,Req7→end),
XeqY(Req43→start,Req8→end), XeqY(Req43→start,Req9→end), XeqY(Req43→start,Req10→end), XeqY(Req43→start,Req11→end),
XeqY(Req43→start,Req12→end), XeqY(Req43→start,Req13→end), XeqY(Req43→start,Req14→end), XeqY(Req43→start,Req15→end),
XeqY(Req43→start,Req16→end), XeqY(Req43→start,Req17→end), XeqY(Req43→start,Req18→end), XeqY(Req43→start,Req19→end),
XeqY(Req43→start,Req20→end), XeqY(Req43→start,Req21→end), XeqY(Req43→start,Req22→end), XeqY(Req43→start,Req23→end),
XeqY(Req43→start,Req24→end), XeqY(Req43→start,Req25→end), XeqY(Req43→start,Req26→end), XeqY(Req43→start,Req27→end),
XeqY(Req43→start,Req28→end), XeqY(Req43→start,Req29→end), XeqY(Req43→start,Req30→end), XeqY(Req43→start,Req31→end),
XeqY(Req43→start,Req32→end), XeqY(Req43→start,Req33→end), XeqY(Req43→start,Req34→end), XeqY(Req43→start,Req35→end),
XeqY(Req43→start,Req36→end), XeqY(Req43→start,Req37→end), XeqY(Req43→start,Req38→end), XeqY(Req43→start,Req39→end),
XeqY(Req43→start,Req40→end), XeqY(Req43→start,Req41→end), XeqY(Req43→start,Req42→end), XeqY(Req43→start,Req44→end),
XeqY(Req43→start,Req45→end), XeqY(Req43→start,Req46→end), XeqY(Req43→start,Req47→end), XeqY(Req43→start,Req48→end),
XeqY(Req43→start,Rel0→start), XeqY(Req43→start,Rel1→start), XeqY(Req43→start,Rel2→start), XeqY(Req43→start, project_end)))

Possible starts of PGR & ADG (GUI-Result): Or(
XeqY(Req44→start,Req0→end), XeqY(Req44→start,Req1→end), XeqY(Req44→start,Req2→end), XeqY(Req44→start,Req3→end),
XeqY(Req44→start,Req4→end), XeqY(Req44→start,Req5→end), XeqY(Req44→start,Req6→end), XeqY(Req44→start,Req7→end),
XeqY(Req44→start,Req8→end), XeqY(Req44→start,Req9→end), XeqY(Req44→start,Req10→end), XeqY(Req44→start,Req11→end),
XeqY(Req44→start,Req12→end), XeqY(Req44→start,Req13→end), XeqY(Req44→start,Req14→end), XeqY(Req44→start,Req15→end),
XeqY(Req44→start,Req16→end), XeqY(Req44→start,Req17→end), XeqY(Req44→start,Req18→end), XeqY(Req44→start,Req19→end),
XeqY(Req44→start,Req20→end), XeqY(Req44→start,Req21→end), XeqY(Req44→start,Req22→end), XeqY(Req44→start,Req23→end),
XeqY(Req44→start,Req24→end), XeqY(Req44→start,Req25→end), XeqY(Req44→start,Req26→end), XeqY(Req44→start,Req27→end),
XeqY(Req44→start,Req28→end), XeqY(Req44→start,Req29→end), XeqY(Req44→start,Req30→end), XeqY(Req44→start,Req31→end),
XeqY(Req44→start,Req32→end), XeqY(Req44→start,Req33→end), XeqY(Req44→start,Req34→end), XeqY(Req44→start,Req35→end),
XeqY(Req44→start,Req36→end), XeqY(Req44→start,Req37→end), XeqY(Req44→start,Req38→end), XeqY(Req44→start,Req39→end),
XeqY(Req44→start,Req40→end), XeqY(Req44→start,Req41→end), XeqY(Req44→start,Req42→end), XeqY(Req44→start,Req43→end),
XeqY(Req44→start,Req45→end), XeqY(Req44→start,Req46→end), XeqY(Req44→start,Req47→end), XeqY(Req44→start,Req48→end),
XeqY(Req44→start,Rel0→start), XeqY(Req44→start,Rel1→start), XeqY(Req44→start,Rel2→start), XeqY(Req44→start, project_end)))

Possible starts of Special Equipment (GUI-results): Or(
XeqY(Req45→start,Req0→end), XeqY(Req45→start,Req1→end), XeqY(Req45→start,Req2→end), XeqY(Req45→start,Req3→end),
XeqY(Req45→start,Req4→end), XeqY(Req45→start,Req5→end), XeqY(Req45→start,Req6→end), XeqY(Req45→start,Req7→end),
XeqY(Req45→start,Req8→end), XeqY(Req45→start,Req9→end), XeqY(Req45→start,Req10→end), XeqY(Req45→start,Req11→end),
XeqY(Req45→start,Req12→end), XeqY(Req45→start,Req13→end), XeqY(Req45→start,Req14→end), XeqY(Req45→start,Req15→end),
XeqY(Req45→start,Req16→end), XeqY(Req45→start,Req17→end), XeqY(Req45→start,Req18→end), XeqY(Req45→start,Req19→end),
XeqY(Req45→start,Req20→end), XeqY(Req45→start,Req21→end), XeqY(Req45→start,Req22→end), XeqY(Req45→start,Req23→end),
XeqY(Req45→start,Req24→end), XeqY(Req45→start,Req25→end), XeqY(Req45→start,Req26→end), XeqY(Req45→start,Req27→end),
XeqY(Req45→start,Req28→end), XeqY(Req45→start,Req29→end), XeqY(Req45→start,Req30→end), XeqY(Req45→start,Req31→end),
XeqY(Req45→start,Req32→end), XeqY(Req45→start,Req33→end), XeqY(Req45→start,Req34→end), XeqY(Req45→start,Req35→end),
XeqY(Req45→start,Req36→end), XeqY(Req45→start,Req37→end), XeqY(Req45→start,Req38→end), XeqY(Req45→start,Req39→end),
XeqY(Req45→start,Req40→end), XeqY(Req45→start,Req41→end), XeqY(Req45→start,Req42→end), XeqY(Req45→start,Req43→end),
XeqY(Req45→start,Req44→end), XeqY(Req45→start,Req46→end), XeqY(Req45→start,Req47→end), XeqY(Req45→start,Req48→end),
XeqY(Req45→start,Rel0→start), XeqY(Req45→start,Rel1→start), XeqY(Req45→start,Rel2→start), XeqY(Req45→start, project_end)))
Possible starts of Special Equipment (Calculation): Or(
XeqY(Req46→start,Req0→end), XeqY(Req46→start,Req1→end), XeqY(Req46→start,Req2→end), XeqY(Req46→start,Req3→end),
XeqY(Req46→start,Req4→end), XeqY(Req46→start,Req5→end), XeqY(Req46→start,Req6→end), XeqY(Req46→start,Req7→end),
XeqY(Req46→start,Req8→end), XeqY(Req46→start,Req9→end), XeqY(Req46→start,Req10→end), XeqY(Req46→start,Req11→end),
XeqY(Req46→start,Req12→end), XeqY(Req46→start,Req13→end), XeqY(Req46→start,Req14→end), XeqY(Req46→start,Req15→end),
XeqY(Req46→start,Req16→end), XeqY(Req46→start,Req17→end), XeqY(Req46→start,Req18→end), XeqY(Req46→start,Req19→end),
XeqY(Req46→start,Req20→end), XeqY(Req46→start,Req21→end), XeqY(Req46→start,Req22→end), XeqY(Req46→start,Req23→end),
XeqY(Req46→start,Req24→end), XeqY(Req46→start,Req25→end), XeqY(Req46→start,Req26→end), XeqY(Req46→start,Req27→end),
XeqY(Req46→start,Req28→end), XeqY(Req46→start,Req29→end), XeqY(Req46→start,Req30→end), XeqY(Req46→start,Req31→end),
XeqY(Req46→start,Req32→end), XeqY(Req46→start,Req33→end), XeqY(Req46→start,Req34→end), XeqY(Req46→start,Req35→end),
XeqY(Req46→start,Req36→end), XeqY(Req46→start,Req37→end), XeqY(Req46→start,Req38→end), XeqY(Req46→start,Req39→end),
XeqY(Req46→start,Req40→end), XeqY(Req46→start,Req41→end), XeqY(Req46→start,Req42→end), XeqY(Req46→start,Req43→end),
XeqY(Req46→start,Req44→end), XeqY(Req46→start,Req45→end), XeqY(Req46→start,Req47→end), XeqY(Req46→start,Req48→end),
XeqY(Req46→start,Rel0→start), XeqY(Req46→start,Rel1→start), XeqY(Req46→start,Rel2→start), XeqY(Req46→start, project_end)))
Possible starts of Eco. efficiency (Graphics): Or(
XeqY(Req47→start,Req0→end), XeqY(Req47→start,Req1→end), XeqY(Req47→start,Req2→end), XeqY(Req47→start,Req3→end),
XeqY(Req47→start,Req4→end), XeqY(Req47→start,Req5→end), XeqY(Req47→start,Req6→end), XeqY(Req47→start,Req7→end),
XeqY(Req47→start,Req8→end), XeqY(Req47→start,Req9→end), XeqY(Req47→start,Req10→end), XeqY(Req47→start,Req11→end),
XeqY(Req47→start,Req12→end), XeqY(Req47→start,Req13→end), XeqY(Req47→start,Req14→end), XeqY(Req47→start,Req15→end),
XeqY(Req47→start,Req16→end), XeqY(Req47→start,Req17→end), XeqY(Req47→start,Req18→end), XeqY(Req47→start,Req19→end),
XeqY(Req47→start,Req20→end), XeqY(Req47→start,Req21→end), XeqY(Req47→start,Req22→end), XeqY(Req47→start,Req23→end),
XeqY(Req47→start,Req24→end), XeqY(Req47→start,Req25→end), XeqY(Req47→start,Req26→end), XeqY(Req47→start,Req27→end),
XeqY(Req47→start,Req28→end), XeqY(Req47→start,Req29→end), XeqY(Req47→start,Req30→end), XeqY(Req47→start,Req31→end),
XeqY(Req47→start,Req32→end), XeqY(Req47→start,Req33→end), XeqY(Req47→start,Req34→end), XeqY(Req47→start,Req35→end),
XeqY(Req47→start,Req36→end), XeqY(Req47→start,Req37→end), XeqY(Req47→start,Req38→end), XeqY(Req47→start,Req39→end),
XeqY(Req47→start,Req40→end), XeqY(Req47→start,Req41→end), XeqY(Req47→start,Req42→end), XeqY(Req47→start,Req43→end),
XeqY(Req47→start,Req44→end), XeqY(Req47→start,Req45→end), XeqY(Req47→start,Req46→end), XeqY(Req47→start,Req48→end),
XeqY(Req47→start,Rel0→start), XeqY(Req47→start,Rel1→start), XeqY(Req47→start,Rel2→start), XeqY(Req47→start, project_end)))
Possible starts of Eco. efficiency (Proposal GUI): Or(
XeqY(Req48→start,Req0→end), XeqY(Req48→start,Req1→end), XeqY(Req48→start,Req2→end), XeqY(Req48→start,Req3→end),
XeqY(Req48→start,Req4→end), XeqY(Req48→start,Req5→end), XeqY(Req48→start,Req6→end), XeqY(Req48→start,Req7→end),
XeqY(Req48→start,Req8→end), XeqY(Req48→start,Req9→end), XeqY(Req48→start,Req10→end), XeqY(Req48→start,Req11→end),
XeqY(Req48→start,Req12→end), XeqY(Req48→start,Req13→end), XeqY(Req48→start,Req14→end), XeqY(Req48→start,Req15→end),
XeqY(Req48→start,Req16→end), XeqY(Req48→start,Req17→end), XeqY(Req48→start,Req18→end), XeqY(Req48→start,Req19→end),
XeqY(Req48→start,Req20→end), XeqY(Req48→start,Req21→end), XeqY(Req48→start,Req22→end), XeqY(Req48→start,Req23→end),
XeqY(Req48→start,Req24→end), XeqY(Req48→start,Req25→end), XeqY(Req48→start,Req26→end), XeqY(Req48→start,Req27→end),
XeqY(Req48→start,Req28→end), XeqY(Req48→start,Req29→end), XeqY(Req48→start,Req30→end), XeqY(Req48→start,Req31→end),
XeqY(Req48→start,Req32→end), XeqY(Req48→start,Req33→end), XeqY(Req48→start,Req34→end), XeqY(Req48→start,Req35→end),
XeqY(Req48→start,Req36→end), XeqY(Req48→start,Req37→end), XeqY(Req48→start,Req38→end), XeqY(Req48→start,Req39→end),
XeqY(Req48→start,Req40→end), XeqY(Req48→start,Req41→end), XeqY(Req48→start,Req42→end), XeqY(Req48→start,Req43→end),
XeqY(Req48→start,Req44→end), XeqY(Req48→start,Req45→end), XeqY(Req48→start,Req46→end), XeqY(Req48→start,Req47→end),
XeqY(Req48→start,Rel0→start), XeqY(Req48→start,Rel1→start), XeqY(Req48→start,Rel2→start), XeqY(Req48→start, project_end)))

Table B.5: Example Application: Start constraints of requirements.

Latest possible release of requirements

'Collectors and installation' is finished at Release1	XlteqC(Req6→assigned_release,0)
'PGR & ADG' is finished at Release2	XlteqC(Req13→assigned_release,1)
'Economic efficiency' is finished at Release3	XlteqC(Req22→assigned_release,2)

Table B.6: Example Application: Latest release of some requirement implementations.

Requirement implemenations that need the same resources

'Auth. Tables' and 'Auth. DB Communication' require the same resource	Or(XlteqY(Req39→end,Req40→start),XgteqY(Req39→start,Req40→end))
'Auth. Tables' and 'Auth. DB Communication' require the same resource	Or(XlteqY(Req39→end,Req40→start),XgteqY(Req39→start,Req40→end))
'Auth. Tables' and 'Auth. Pwd Encryption' require the same resource	Or(XlteqY(Req39→end,Req41→start),XgteqY(Req39→start,Req41→end))
'Auth. Tables' and 'Auth. Pwd Encryption' require the same resource	Or(XlteqY(Req39→end,Req41→start),XgteqY(Req39→start,Req41→end))
'Auth. DB Communication' and 'Auth. Tables' require the same resource	Or(XlteqY(Req40→end,Req39→start),XgteqY(Req40→start,Req39→end))
'Auth. DB Communication' and 'Auth. Tables' require the same resource	Or(XlteqY(Req40→end,Req39→start),XgteqY(Req40→start,Req39→end))
'Auth. DB Communication' and 'Auth. Pwd Encryption' require the same resource	Or(XlteqY(Req40→end,Req41→start),XgteqY(Req40→start,Req41→end))
'Auth. DB Communication' and 'Auth. Pwd Encryption' require the same resource	Or(XlteqY(Req40→end,Req41→start),XgteqY(Req40→start,Req41→end))
'Auth. Pwd Encryption' and 'Auth. Tables' require the same resource	Or(XlteqY(Req41→end,Req39→start),XgteqY(Req41→start,Req39→end))
'Auth. Pwd Encryption' and 'Auth. Tables' require the same resource	Or(XlteqY(Req41→end,Req39→start),XgteqY(Req41→start,Req39→end))
'Auth. Pwd Encryption' and 'Auth. DB Communication' require the same resource	Or(XlteqY(Req41→end,Req40→start),XgteqY(Req41→start,Req40→end))
'Auth. Pwd Encryption' and 'Auth. DB Communication' require the same resource	Or(XlteqY(Req41→end,Req40→start),XgteqY(Req41→start,Req40→end))

Table B.7: Example Application: Requirement implementations with same resources.

Bounded, needed, and used resources

'Multilingualism'	XplusYeqZ(Req0→bounded_developers,Req0→needed_developers,Req0→used_developers) XplusYeqZ(Req0→bounded_tester,Req0→needed_tester,Req0→used_tester) XplusYeqZ(Req0→bounded_budget,Req0→needed_budget,Req0→used_budget)
'Authentication'	XplusYeqZ(Req1→bounded_developers,Req1→needed_developers,Req1→used_developers) XplusYeqZ(Req1→bounded_tester,Req1→needed_tester,Req1→used_tester) XplusYeqZ(Req1→bounded_budget,Req1→needed_budget,Req1→used_budget)
'Collectors' surface'	XplusYeqZ(Req2→bounded_developers,Req2→needed_developers,Req2→used_developers) XplusYeqZ(Req2→bounded_tester,Req2→needed_tester,Req2→used_tester) XplusYeqZ(Req2→bounded_budget,Req2→needed_budget,Req2→used_budget)
'GUI Preconditions'	XplusYeqZ(Req3→bounded_developers,Req3→needed_developers,Req3→used_developers) XplusYeqZ(Req3→bounded_tester,Req3→needed_tester,Req3→used_tester) XplusYeqZ(Req3→bounded_budget,Req3→needed_budget,Req3→used_budget)
'Collectors' surface (GUI)'	XplusYeqZ(Req4→bounded_developers,Req4→needed_developers,Req4→used_developers) XplusYeqZ(Req4→bounded_tester,Req4→needed_tester,Req4→used_tester) XplusYeqZ(Req4→bounded_budget,Req4→needed_budget,Req4→used_budget)
'Database Backend'	XplusYeqZ(Req5→bounded_developers,Req5→needed_developers,Req5→used_developers) XplusYeqZ(Req5→bounded_tester,Req5→needed_tester,Req5→used_tester) XplusYeqZ(Req5→bounded_budget,Req5→needed_budget,Req5→used_budget)
'Collectors and installation'	XplusYeqZ(Req6→bounded_developers,Req6→needed_developers,Req6→used_developers) XplusYeqZ(Req6→bounded_tester,Req6→needed_tester,Req6→used_tester) XplusYeqZ(Req6→bounded_budget,Req6→needed_budget,Req6→used_budget)
'Collectors's surface (Store)'	XplusYeqZ(Req7→bounded_developers,Req7→needed_developers,Req7→used_developers) XplusYeqZ(Req7→bounded_tester,Req7→needed_tester,Req7→used_tester) XplusYeqZ(Req7→bounded_budget,Req7→needed_budget,Req7→used_budget)
'Collectors and installation (Store)'	XplusYeqZ(Req8→bounded_developers,Req8→needed_developers,Req8→used_developers) XplusYeqZ(Req8→bounded_tester,Req8→needed_tester,Req8→used_tester) XplusYeqZ(Req8→bounded_budget,Req8→needed_budget,Req8→used_budget)
'Air caps calculation'	XplusYeqZ(Req9→bounded_developers,Req9→needed_developers,Req9→used_developers) XplusYeqZ(Req9→bounded_tester,Req9→needed_tester,Req9→used_tester) XplusYeqZ(Req9→bounded_budget,Req9→needed_budget,Req9→used_budget)
'Air caps calculation (Store)'	XplusYeqZ(Req10→bounded_developers,Req10→needed_developers,Req10→used_developers) XplusYeqZ(Req10→bounded_tester,Req10→needed_tester,Req10→used_tester) XplusYeqZ(Req10→bounded_budget,Req10→needed_budget,Req10→used_budget)
'Pipe Costs'	XplusYeqZ(Req11→bounded_developers,Req11→needed_developers,Req11→used_developers) XplusYeqZ(Req11→bounded_tester,Req11→needed_tester,Req11→used_tester) XplusYeqZ(Req11→bounded_budget,Req11→needed_budget,Req11→used_budget)
'Pipe Costs (Store)'	XplusYeqZ(Req12→bounded_developers,Req12→needed_developers,Req12→used_developers) XplusYeqZ(Req12→bounded_tester,Req12→needed_tester,Req12→used_tester) XplusYeqZ(Req12→bounded_budget,Req12→needed_budget,Req12→used_budget)
'PGR & ADG'	XplusYeqZ(Req13→bounded_developers,Req13→needed_developers,Req13→used_developers) XplusYeqZ(Req13→bounded_tester,Req13→needed_tester,Req13→used_tester) XplusYeqZ(Req13→bounded_budget,Req13→needed_budget,Req13→used_budget)
'PGR & ADG (Store)'	XplusYeqZ(Req14→bounded_developers,Req14→needed_developers,Req14→used_developers) XplusYeqZ(Req14→bounded_tester,Req14→needed_tester,Req14→used_tester) XplusYeqZ(Req14→bounded_budget,Req14→needed_budget,Req14→used_budget)
'Costs for refilling'	XplusYeqZ(Req15→bounded_developers,Req15→needed_developers,Req15→used_developers) XplusYeqZ(Req15→bounded_tester,Req15→needed_tester,Req15→used_tester) XplusYeqZ(Req15→bounded_budget,Req15→needed_budget,Req15→used_budget)
'Costs for refilling (Store)'	XplusYeqZ(Req16→bounded_developers,Req16→needed_developers,Req16→used_developers) XplusYeqZ(Req16→bounded_tester,Req16→needed_tester,Req16→used_tester) XplusYeqZ(Req16→bounded_budget,Req16→needed_budget,Req16→used_budget)
'Special Equipment'	XplusYeqZ(Req17→bounded_developers,Req17→needed_developers,Req17→used_developers)

	XplusYeqZ(Req17→bounded_tester,Req17→needed_tester,Req17→used_tester)
	XplusYeqZ(Req17→bounded_budget,Req17→needed_budget,Req17→used_budget)
'Special Equipment (Store)'	XplusYeqZ(Req18→bounded_developers,Req18→needed_developers,Req18→used_developers)
	XplusYeqZ(Req18→bounded_tester,Req18→needed_tester,Req18→used_tester)
	XplusYeqZ(Req18→bounded_budget,Req18→needed_budget,Req18→used_budget)
'Misc. Material'	XplusYeqZ(Req19→bounded_developers,Req19→needed_developers,Req19→used_developers)
	XplusYeqZ(Req19→bounded_tester,Req19→needed_tester,Req19→used_tester)
	XplusYeqZ(Req19→bounded_budget,Req19→needed_budget,Req19→used_budget)
'Misc. Material (Store)'	XplusYeqZ(Req20→bounded_developers,Req20→needed_developers,Req20→used_developers)
	XplusYeqZ(Req20→bounded_tester,Req20→needed_tester,Req20→used_tester)
	XplusYeqZ(Req20→bounded_budget,Req20→needed_budget,Req20→used_budget)
'Misc. Material (GUI)'	XplusYeqZ(Req21→bounded_developers,Req21→needed_developers,Req21→used_developers)
	XplusYeqZ(Req21→bounded_tester,Req21→needed_tester,Req21→used_tester)
	XplusYeqZ(Req21→bounded_budget,Req21→needed_budget,Req21→used_budget)
'Economic efficiency'	XplusYeqZ(Req22→bounded_developers,Req22→needed_developers,Req22→used_developers)
	XplusYeqZ(Req22→bounded_tester,Req22→needed_tester,Req22→used_tester)
	XplusYeqZ(Req22→bounded_budget,Req22→needed_budget,Req22→used_budget)
'Collectors and installation (GUI)'	XplusYeqZ(Req23→bounded_developers,Req23→needed_developers,Req23→used_developers)
	XplusYeqZ(Req23→bounded_tester,Req23→needed_tester,Req23→used_tester)
	XplusYeqZ(Req23→bounded_budget,Req23→needed_budget,Req23→used_budget)
'Pipe costs (GUI-input)'	XplusYeqZ(Req24→bounded_developers,Req24→needed_developers,Req24→used_developers)
	XplusYeqZ(Req24→bounded_tester,Req24→needed_tester,Req24→used_tester)
	XplusYeqZ(Req24→bounded_budget,Req24→needed_budget,Req24→used_budget)
'Air caps calculation (GUI-Input)'	XplusYeqZ(Req25→bounded_developers,Req25→needed_developers,Req25→used_developers)
	XplusYeqZ(Req25→bounded_tester,Req25→needed_tester,Req25→used_tester)
	XplusYeqZ(Req25→bounded_budget,Req25→needed_budget,Req25→used_budget)
'PGR & ADG (GUI-Input)'	XplusYeqZ(Req26→bounded_developers,Req26→needed_developers,Req26→used_developers)
	XplusYeqZ(Req26→bounded_tester,Req26→needed_tester,Req26→used_tester)
	XplusYeqZ(Req26→bounded_budget,Req26→needed_budget,Req26→used_budget)
'Costs for refilling (Calculation)'	XplusYeqZ(Req27→bounded_developers,Req27→needed_developers,Req27→used_developers)
	XplusYeqZ(Req27→bounded_tester,Req27→needed_tester,Req27→used_tester)
	XplusYeqZ(Req27→bounded_budget,Req27→needed_budget,Req27→used_budget)
'Special Equipment (GUI-input)'	XplusYeqZ(Req28→bounded_developers,Req28→needed_developers,Req28→used_developers)
	XplusYeqZ(Req28→bounded_tester,Req28→needed_tester,Req28→used_tester)
	XplusYeqZ(Req28→bounded_budget,Req28→needed_budget,Req28→used_budget)
'PGR & ADG (Calculation) '	XplusYeqZ(Req29→bounded_developers,Req29→needed_developers,Req29→used_developers)
	XplusYeqZ(Req29→bounded_tester,Req29→needed_tester,Req29→used_tester)
	XplusYeqZ(Req29→bounded_budget,Req29→needed_budget,Req29→used_budget)
'Eco. efficiency (Calc.)'	XplusYeqZ(Req30→bounded_developers,Req30→needed_developers,Req30→used_developers)
	XplusYeqZ(Req30→bounded_tester,Req30→needed_tester,Req30→used_tester)
	XplusYeqZ(Req30→bounded_budget,Req30→needed_budget,Req30→used_budget)
'Eco. efficiency (GUI-input)'	XplusYeqZ(Req31→bounded_developers,Req31→needed_developers,Req31→used_developers)
	XplusYeqZ(Req31→bounded_tester,Req31→needed_tester,Req31→used_tester)
	XplusYeqZ(Req31→bounded_budget,Req31→needed_budget,Req31→used_budget)
'Eco. efficiency (Proposal)'	XplusYeqZ(Req32→bounded_developers,Req32→needed_developers,Req32→used_developers)
	XplusYeqZ(Req32→bounded_tester,Req32→needed_tester,Req32→used_tester)
	XplusYeqZ(Req32→bounded_budget,Req32→needed_budget,Req32→used_budget)
'Modelling an Iconset'	XplusYeqZ(Req33→bounded_developers,Req33→needed_developers,Req33→used_developers)
	XplusYeqZ(Req33→bounded_tester,Req33→needed_tester,Req33→used_tester)
	XplusYeqZ(Req33→bounded_budget,Req33→needed_budget,Req33→used_budget)
'Mysql Backend'	XplusYeqZ(Req34→bounded_developers,Req34→needed_developers,Req34→used_developers)
	XplusYeqZ(Req34→bounded_tester,Req34→needed_tester,Req34→used_tester)
	XplusYeqZ(Req34→bounded_budget,Req34→needed_budget,Req34→used_budget)
'Sqlite Backend'	XplusYeqZ(Req35→bounded_developers,Req35→needed_developers,Req35→used_developers)

	XplusYeqZ(Req35→bounded_tester,Req35→needed_tester,Req35→used_tester)
	XplusYeqZ(Req35→bounded_budget,Req35→needed_budget,Req35→used_budget)
'Costs for refilling (GUI-input)'	XplusYeqZ(Req36→bounded_developers,Req36→needed_developers,Req36→used_developers)
	XplusYeqZ(Req36→bounded_tester,Req36→needed_tester,Req36→used_tester)
	XplusYeqZ(Req36→bounded_budget,Req36→needed_budget,Req36→used_budget)
'Apache Authentication'	XplusYeqZ(Req37→bounded_developers,Req37→needed_developers,Req37→used_developers)
	XplusYeqZ(Req37→bounded_tester,Req37→needed_tester,Req37→used_tester)
	XplusYeqZ(Req37→bounded_budget,Req37→needed_budget,Req37→used_budget)
'DB Auth.'	XplusYeqZ(Req38→bounded_developers,Req38→needed_developers,Req38→used_developers)
	XplusYeqZ(Req38→bounded_tester,Req38→needed_tester,Req38→used_tester)
	XplusYeqZ(Req38→bounded_budget,Req38→needed_budget,Req38→used_budget)
'Auth. Tables'	XplusYeqZ(Req39→bounded_developers,Req39→needed_developers,Req39→used_developers)
	XplusYeqZ(Req39→bounded_tester,Req39→needed_tester,Req39→used_tester)
	XplusYeqZ(Req39→bounded_budget,Req39→needed_budget,Req39→used_budget)
'Auth. DB Communication'	XplusYeqZ(Req40→bounded_developers,Req40→needed_developers,Req40→used_developers)
	XplusYeqZ(Req40→bounded_tester,Req40→needed_tester,Req40→used_tester)
	XplusYeqZ(Req40→bounded_budget,Req40→needed_budget,Req40→used_budget)
'Auth. Pwd Encryption'	XplusYeqZ(Req41→bounded_developers,Req41→needed_developers,Req41→used_developers)
	XplusYeqZ(Req41→bounded_tester,Req41→needed_tester,Req41→used_tester)
	XplusYeqZ(Req41→bounded_budget,Req41→needed_budget,Req41→used_budget)
'Air caps calculation (GUI-Results)'	XplusYeqZ(Req42→bounded_developers,Req42→needed_developers,Req42→used_developers)
	XplusYeqZ(Req42→bounded_tester,Req42→needed_tester,Req42→used_tester)
	XplusYeqZ(Req42→bounded_budget,Req42→needed_budget,Req42→used_budget)
'Pipe costs (GUI-results)'	XplusYeqZ(Req43→bounded_developers,Req43→needed_developers,Req43→used_developers)
	XplusYeqZ(Req43→bounded_tester,Req43→needed_tester,Req43→used_tester)
	XplusYeqZ(Req43→bounded_budget,Req43→needed_budget,Req43→used_budget)
'PGR & ADG (GUI-Result)'	XplusYeqZ(Req44→bounded_developers,Req44→needed_developers,Req44→used_developers)
	XplusYeqZ(Req44→bounded_tester,Req44→needed_tester,Req44→used_tester)
	XplusYeqZ(Req44→bounded_budget,Req44→needed_budget,Req44→used_budget)
'Special Equipment (GUI-results)'	XplusYeqZ(Req45→bounded_developers,Req45→needed_developers,Req45→used_developers)
	XplusYeqZ(Req45→bounded_tester,Req45→needed_tester,Req45→used_tester)
	XplusYeqZ(Req45→bounded_budget,Req45→needed_budget,Req45→used_budget)
'Special Equipment (Calculation)'	XplusYeqZ(Req46→bounded_developers,Req46→needed_developers,Req46→used_developers)
	XplusYeqZ(Req46→bounded_tester,Req46→needed_tester,Req46→used_tester)
	XplusYeqZ(Req46→bounded_budget,Req46→needed_budget,Req46→used_budget)
'Eco. efficiency (Graphics)'	XplusYeqZ(Req47→bounded_developers,Req47→needed_developers,Req47→used_developers)
	XplusYeqZ(Req47→bounded_tester,Req47→needed_tester,Req47→used_tester)
	XplusYeqZ(Req47→bounded_budget,Req47→needed_budget,Req47→used_budget)
'Eco. efficiency (Proposal GUI)'	XplusYeqZ(Req48→bounded_developers,Req48→needed_developers,Req48→used_developers)
	XplusYeqZ(Req48→bounded_tester,Req48→needed_tester,Req48→used_tester)
	XplusYeqZ(Req48→bounded_budget,Req48→needed_budget,Req48→used_budget)

Table B.8: Example Application: Relations between bounded, needed, and used resources.

Diff2 constraint for the resources

'devel'	Diff2(store, {Req0→start, Req1→start, Req2→start, Req3→start, Req4→start, Req5→start, Req6→start, Req7→start, Req8→start, Req9→start, Req10→start, Req11→start, Req12→start, Req13→start, Req14→start, Req15→start, Req16→start, Req17→start, Req18→start, Req19→start, Req20→start, Req21→start, Req22→start, Req23→start, Req24→start, Req25→start, Req26→start, Req27→start, Req28→start, Req29→start, Req30→start, Req31→start, Req32→start, Req33→start, Req34→start, Req35→start, Req36→start, Req37→start, Req38→start, Req39→start, Req40→start, Req41→start, Req42→start, Req43→start, Req44→start, Req45→start, Req46→start, Req47→start, Req48→start} {Req0→bounded_devel, Req1→bounded_devel, Req2→bounded_devel, Req3→bounded_devel, Req4→bounded_devel, Req5→bounded_devel, Req6→bounded_devel, Req7→bounded_devel, Req8→bounded_devel, Req9→bounded_devel, Req10→bounded_devel, Req11→bounded_devel, Req12→bounded_devel, Req13→bounded_devel, Req14→bounded_devel, Req15→bounded_devel, Req16→bounded_devel, Req17→bounded_devel, Req18→bounded_devel, Req19→bounded_devel, Req20→bounded_devel, Req21→bounded_devel, Req22→bounded_devel, Req23→bounded_devel, Req24→bounded_devel, Req25→bounded_devel, Req26→bounded_devel, Req27→bounded_devel, Req28→bounded_devel, Req29→bounded_devel, Req30→bounded_devel, Req31→bounded_devel, Req32→bounded_devel, Req33→bounded_devel, Req34→bounded_devel, Req35→bounded_devel, Req36→bounded_devel, Req37→bounded_devel, Req38→bounded_devel, Req39→bounded_devel, Req40→bounded_devel, Req41→bounded_devel, Req42→bounded_devel, Req43→bounded_devel, Req44→bounded_devel, Req45→bounded_devel, Req46→bounded_devel, Req47→bounded_devel, Req48→bounded_devel} {Req0→duration, Req1→duration, Req2→duration, Req3→duration, Req4→duration, Req5→duration, Req6→duration, Req7→duration, Req8→duration, Req9→duration, Req10→duration, Req11→duration, Req12→duration, Req13→duration, Req14→duration, Req15→duration, Req16→duration, Req17→duration, Req18→duration, Req19→duration, Req20→duration, Req21→duration, Req22→duration, Req23→duration, Req24→duration, Req25→duration, Req26→duration, Req27→duration, Req28→duration, Req29→duration, Req30→duration, Req31→duration, Req32→duration, Req33→duration, Req34→duration, Req35→duration, Req36→duration, Req37→duration, Req38→duration, Req39→duration, Req40→duration, Req41→duration, Req42→duration, Req43→duration, Req44→duration, Req45→duration, Req46→duration, Req47→duration, Req48→duration} {Req0→needed_devel, Req1→needed_devel, Req2→needed_devel, Req3→needed_devel, Req4→needed_devel, Req5→needed_devel, Req6→needed_devel, Req7→needed_devel, Req8→needed_devel, Req9→needed_devel, Req10→needed_devel, Req11→needed_devel, Req12→needed_devel, Req13→needed_devel, Req14→needed_devel, Req15→needed_devel, Req16→needed_devel, Req17→needed_devel, Req18→needed_devel, Req19→needed_devel, Req20→needed_devel, Req21→needed_devel, Req22→needed_devel, Req23→needed_devel, Req24→needed_devel, Req25→needed_devel, Req26→needed_devel, Req27→needed_devel, Req28→needed_devel, Req29→needed_devel, Req30→needed_devel, Req31→needed_devel, Req32→needed_devel, Req33→needed_devel, Req34→needed_devel, Req35→needed_devel, Req36→needed_devel, Req37→needed_devel, Req38→needed_devel, Req39→needed_devel, Req40→needed_devel, Req41→needed_devel, Req42→needed_devel, Req43→needed_devel, Req44→needed_devel, Req45→needed_devel, Req46→needed_devel, Req47→needed_devel, Req48→needed_devel}
'testers'	Diff2(store, {Req0→start, Req1→start, Req2→start, Req3→start, Req4→start, Req5→start, Req6→start, Req7→start, Req8→start, Req9→start, Req10→start, Req11→start, Req12→start, Req13→start, Req14→start, Req15→start, Req16→start, Req17→start, Req18→start, Req19→start, Req20→start, Req21→start, Req22→start, Req23→start, Req24→start, Req25→start, Req26→start, Req27→start, Req28→start, Req29→start, Req30→start, Req31→start, Req32→start, Req33→start, Req34→start, Req35→start, Req36→start, Req37→start, Req38→start, Req39→start, Req40→start, Req41→start, Req42→start, Req43→start, Req44→start, Req45→start, Req46→start, Req47→start, Req48→start} {Req0→bounded_testers, Req1→bounded_testers, Req2→bounded_testers, Req3→bounded_testers, Req4→bounded_testers, Req5→bounded_testers, Req6→bounded_testers, Req7→bounded_testers, Req8→bounded_testers, Req9→bounded_testers, Req10→bounded_testers, Req11→bounded_testers, Req12→bounded_testers, Req13→bounded_testers, Req14→bounded_testers, Req15→bounded_testers, Req16→bounded_testers, Req17→bounded_testers, Req18→bounded_testers, Req19→bounded_testers, Req20→bounded_testers, Req21→bounded_testers, Req22→bounded_testers, Req23→bounded_testers, Req24→bounded_testers, Req25→bounded_testers, Req26→bounded_testers, Req27→bounded_testers, Req28→bounded_testers, Req29→bounded_testers, Req30→bounded_testers, Req31→bounded_testers, Req32→bounded_testers, Req33→bounded_testers, Req34→bounded_testers, Req35→bounded_testers, Req36→bounded_testers, Req37→bounded_testers, Req38→bounded_testers, Req39→bounded_testers, Req40→bounded_testers, Req41→bounded_testers, Req42→bounded_testers, Req43→bounded_testers, Req44→bounded_testers, Req45→bounded_testers, Req46→bounded_testers, Req47→bounded_testers, Req48→bounded_testers} {Req0→duration, Req1→duration, Req2→duration, Req3→duration, Req4→duration, Req5→duration, Req6→duration, Req7→duration, Req8→duration, Req9→duration, Req10→duration, Req11→duration, Req12→duration, Req13→duration, Req14→duration, Req15→duration, Req16→duration, Req17→duration, Req18→duration, Req19→duration, Req20→duration, Req21→duration, Req22→duration, Req23→duration, Req24→duration, Req25→duration, Req26→duration, Req27→duration, Req28→duration, Req29→duration, Req30→duration, Req31→duration, Req32→duration, Req33→duration, Req34→duration, Req35→duration, Req36→duration, Req37→duration, Req38→duration, Req39→duration, Req40→duration, Req41→duration, Req42→duration, Req43→duration, Req44→duration, Req45→duration, Req46→duration, Req47→duration, Req48→duration}

	{Req0→needed_testers, Req1→needed_testers, Req2→needed_testers, Req3→needed_testers, Req4→needed_testers, Req5→needed_testers, Req6→needed_testers, Req7→needed_testers, Req8→needed_testers, Req9→needed_testers, Req10→needed_testers, Req11→needed_testers, Req12→needed_testers, Req13→needed_testers, Req14→needed_testers, Req15→needed_testers, Req16→needed_testers, Req17→needed_testers, Req18→needed_testers, Req19→needed_testers, Req20→needed_testers, Req21→needed_testers, Req22→needed_testers, Req23→needed_testers, Req24→needed_testers, Req25→needed_testers, Req26→needed_testers, Req27→needed_testers, Req28→needed_testers, Req29→needed_testers, Req30→needed_testers, Req31→needed_testers, Req32→needed_testers, Req33→needed_testers, Req34→needed_testers, Req35→needed_testers, Req36→needed_testers, Req37→needed_testers, Req38→needed_testers, Req39→needed_testers, Req40→needed_testers, Req41→needed_testers, Req42→needed_testers, Req43→needed_testers, Req44→needed_testers, Req45→needed_testers, Req46→needed_testers, Req47→needed_testers, Req48→needed_testers}
'Budget'	Diff2(store, {Req0→release_start, Req1→release_start, Req2→release_start, Req3→release_start, Req4→release_start, Req5→release_start, Req6→release_start, Req7→release_start, Req8→release_start, Req9→release_start, Req10→release_start, Req11→release_start, Req12→release_start, Req13→release_start, Req14→release_start, Req15→release_start, Req16→release_start, Req17→release_start, Req18→release_start, Req19→release_start, Req20→release_start, Req21→release_start, Req22→release_start, Req23→release_start, Req24→release_start, Req25→release_start, Req26→release_start, Req27→release_start, Req28→release_start, Req29→release_start, Req30→release_start, Req31→release_start, Req32→release_start, Req33→release_start, Req34→release_start, Req35→release_start, Req36→release_start, Req37→release_start, Req38→release_start, Req39→release_start, Req40→release_start, Req41→release_start, Req42→release_start, Req43→release_start, Req44→release_start, Req45→release_start, Req46→release_start, Req47→release_start, Req48→release_start} {Req0→bounded_budget, Req1→bounded_budget, Req2→bounded_budget, Req3→bounded_budget, Req4→bounded_budget, Req5→bounded_budget, Req6→bounded_budget, Req7→bounded_budget, Req8→bounded_budget, Req9→bounded_budget, Req10→bounded_budget, Req11→bounded_budget, Req12→bounded_budget, Req13→bounded_budget, Req14→bounded_budget, Req15→bounded_budget, Req16→bounded_budget, Req17→bounded_budget, Req18→bounded_budget, Req19→bounded_budget, Req20→bounded_budget, Req21→bounded_budget, Req22→bounded_budget, Req23→bounded_budget, Req24→bounded_budget, Req25→bounded_budget, Req26→bounded_budget, Req27→bounded_budget, Req28→bounded_budget, Req29→bounded_budget, Req30→bounded_budget, Req31→bounded_budget, Req32→bounded_budget, Req33→bounded_budget, Req34→bounded_budget, Req35→bounded_budget, Req36→bounded_budget, Req37→bounded_budget, Req38→bounded_budget, Req39→bounded_budget, Req40→bounded_budget, Req41→bounded_budget, Req42→bounded_budget, Req43→bounded_budget, Req44→bounded_budget, Req45→bounded_budget, Req46→bounded_budget, Req47→bounded_budget, Req48→bounded_budget} {Req0→release_duration, Req1→release_duration, Req2→release_duration, Req3→release_duration, Req4→release_duration, Req5→release_duration, Req6→release_duration, Req7→release_duration, Req8→release_duration, Req9→release_duration, Req10→release_duration, Req11→release_duration, Req12→release_duration, Req13→release_duration, Req14→release_duration, Req15→release_duration, Req16→release_duration, Req17→release_duration, Req18→release_duration, Req19→release_duration, Req20→release_duration, Req21→release_duration, Req22→release_duration, Req23→release_duration, Req24→release_duration, Req25→release_duration, Req26→release_duration, Req27→release_duration, Req28→release_duration, Req29→release_duration, Req30→release_duration, Req31→release_duration, Req32→release_duration, Req33→release_duration, Req34→release_duration, Req35→release_duration, Req36→release_duration, Req37→release_duration, Req38→release_duration, Req39→release_duration, Req40→release_duration, Req41→release_duration, Req42→release_duration, Req43→release_duration, Req44→release_duration, Req45→release_duration, Req46→release_duration, Req47→release_duration, Req48→release_duration} {Req0→needed_budget, Req1→needed_budget, Req2→needed_budget, Req3→needed_budget, Req4→needed_budget, Req5→needed_budget, Req6→needed_budget, Req7→needed_budget, Req8→needed_budget, Req9→needed_budget, Req10→needed_budget, Req11→needed_budget, Req12→needed_budget, Req13→needed_budget, Req14→needed_budget, Req15→needed_budget, Req16→needed_budget, Req17→needed_budget, Req18→needed_budget, Req19→needed_budget, Req20→needed_budget, Req21→needed_budget, Req22→needed_budget, Req23→needed_budget, Req24→needed_budget, Req25→needed_budget, Req26→needed_budget, Req27→needed_budget, Req28→needed_budget, Req29→needed_budget, Req30→needed_budget, Req31→needed_budget, Req32→needed_budget, Req33→needed_budget, Req34→needed_budget, Req35→needed_budget, Req36→needed_budget, Req37→needed_budget, Req38→needed_budget, Req39→needed_budget, Req40→needed_budget, Req41→needed_budget, Req42→needed_budget, Req43→needed_budget, Req44→needed_budget, Req45→needed_budget, Req46→needed_budget, Req47→needed_budget, Req48→needed_budget}

Table B.9: Example Application: Diff2 constraint for developers, testers, and budget.

Release dependent constraints

'Multilingualism' in Rel0: IfThenElse(And(XgteqY(Req0→start,Rel0→start), XlteqY(Req0→end,Rel0→end)), And(XlteqC(Req0→used_devel,3), XlteqC(Req0→used_testers,2), XlteqC(Req0→used_budget,10000), XeqY(Req0→release_duration,Rel0→duration), XeqY(Req0→release_start,Rel0→start), XeqC(Req0→assigned_release,0)), XneqC(Req0→assigned_release,0))

'Authentication' in Rel0: IfThenElse(And(XgteqY(Req1→start,Rel0→start), XlteqY(Req1→end,Rel0→end)), And(XlteqC(Req1→used_devel,3), XlteqC(Req1→used_testers,2), XlteqC(Req1→used_budget,10000), XeqY(Req1→release_duration,Rel0→duration), XeqY(Req1→release_start,Rel0→start), XeqC(Req1→assigned_release,0)), XneqC(Req1→assigned_release,0))

'Collectors' surface' in Rel0: IfThenElse(And(XgteqY(Req2→start,Rel0→start), XlteqY(Req2→end,Rel0→end)), And(XlteqC(Req2→used_devel,3), XlteqC(Req2→used_testers,2), XlteqC(Req2→used_budget,10000), XeqY(Req2→release_duration,Rel0→duration), XeqY(Req2→release_start,Rel0→start), XeqC(Req2→assigned_release,0)), XneqC(Req2→assigned_release,0))

'GUI Preconditions' in Rel0: IfThenElse(And(XgteqY(Req3→start,Rel0→start), XlteqY(Req3→end,Rel0→end)), And(XlteqC(Req3→used_devel,3), XlteqC(Req3→used_testers,2), XlteqC(Req3→used_budget,10000), XeqY(Req3→release_duration,Rel0→duration), XeqY(Req3→release_start,Rel0→start), XeqC(Req3→assigned_release,0)), XneqC(Req3→assigned_release,0))

'Collectors' surface (GUI)' in Rel0: IfThenElse(And(XgteqY(Req4→start,Rel0→start), XlteqY(Req4→end,Rel0→end)), And(XlteqC(Req4→used_devel,3), XlteqC(Req4→used_testers,2), XlteqC(Req4→used_budget,10000), XeqY(Req4→release_duration,Rel0→duration), XeqY(Req4→release_start,Rel0→start), XeqC(Req4→assigned_release,0)), XneqC(Req4→assigned_release,0))

'Database Backend' in Rel0: IfThenElse(And(XgteqY(Req5→start,Rel0→start), XlteqY(Req5→end,Rel0→end)), And(XlteqC(Req5→used_devel,3), XlteqC(Req5→used_testers,2), XlteqC(Req5→used_budget,10000), XeqY(Req5→release_duration,Rel0→duration), XeqY(Req5→release_start,Rel0→start), XeqC(Req5→assigned_release,0)), XneqC(Req5→assigned_release,0))

'Collectors and installation' in Rel0: IfThenElse(And(XgteqY(Req6→start,Rel0→start), XlteqY(Req6→end,Rel0→end)), And(XlteqC(Req6→used_devel,3), XlteqC(Req6→used_testers,2), XlteqC(Req6→used_budget,10000), XeqY(Req6→release_duration,Rel0→duration), XeqY(Req6→release_start,Rel0→start), XeqC(Req6→assigned_release,0)), XneqC(Req6→assigned_release,0))

'Collectors's surface (Store)' in Rel0: IfThenElse(And(XgteqY(Req7→start,Rel0→start), XlteqY(Req7→end,Rel0→end)), And(XlteqC(Req7→used_devel,3), XlteqC(Req7→used_testers,2), XlteqC(Req7→used_budget,10000), XeqY(Req7→release_duration,Rel0→duration), XeqY(Req7→release_start,Rel0→start), XeqC(Req7→assigned_release,0)), XneqC(Req7→assigned_release,0))

'Collectors and installation (Store)' in Rel0: IfThenElse(And(XgteqY(Req8→start,Rel0→start), XlteqY(Req8→end,Rel0→end)), And(XlteqC(Req8→used_devel,3), XlteqC(Req8→used_testers,2), XlteqC(Req8→used_budget,10000), XeqY(Req8→release_duration,Rel0→duration), XeqY(Req8→release_start,Rel0→start), XeqC(Req8→assigned_release,0)), XneqC(Req8→assigned_release,0))

'Air caps calculation' in Rel0: IfThenElse(And(XgteqY(Req9→start,Rel0→start), XlteqY(Req9→end,Rel0→end)), And(XlteqC(Req9→used_devel,3), XlteqC(Req9→used_testers,2), XlteqC(Req9→used_budget,10000), XeqY(Req9→release_duration,Rel0→duration), XeqY(Req9→release_start,Rel0→start), XeqC(Req9→assigned_release,0)), XneqC(Req9→assigned_release,0))

'Air caps calculation (Store)' in Rel0: IfThenElse(And(XgteqY(Req10→start,Rel0→start), XlteqY(Req10→end,Rel0→end)), And(XlteqC(Req10→used_devel,3), XlteqC(Req10→used_testers,2), XlteqC(Req10→used_budget,10000), XeqY(Req10→release_duration,Rel0→duration), XeqY(Req10→release_start,Rel0→start), XeqC(Req10→assigned_release,0)), XneqC(Req10→assigned_release,0))

'Pipe Costs' in Rel0: IfThenElse(And(XgteqY(Req11→start,Rel0→start), XlteqY(Req11→end,Rel0→end)), And(XlteqC(Req11→used_devel,3), XlteqC(Req11→used_testers,2), XlteqC(Req11→used_budget,10000), XeqY(Req11→release_duration,Rel0→duration), XeqY(Req11→release_start,Rel0→start), XeqC(Req11→assigned_release,0)), XneqC(Req11→assigned_release,0))

'Pipe Costs (Store)' in Rel0: IfThenElse(And(XgteqY(Req12→start,Rel0→start), XlteqY(Req12→end,Rel0→end)), And(XlteqC(Req12→used_devel,3), XlteqC(Req12→used_testers,2), XlteqC(Req12→used_budget,10000), XeqY(Req12→release_duration,Rel0→duration), XeqY(Req12→release_start,Rel0→start), XeqC(Req12→assigned_release,0)), XneqC(Req12→assigned_release,0))

'PGR & ADG' in Rel0: IfThenElse(And(XgteqY(Req13→start,Rel0→start), XlteqY(Req13→end,Rel0→end)), And(XlteqC(Req13→used_devel,3), XlteqC(Req13→used_testers,2), XlteqC(Req13→used_budget,10000), XeqY(Req13→release_duration,Rel0→duration), XeqY(Req13→release_start,Rel0→start), XeqC(Req13→assigned_release,0)), XneqC(Req13→assigned_release,0))

'PGR & ADG (Store)' in Rel0: IfThenElse(And(XgteqY(Req14→start,Rel0→start), XlteqY(Req14→end,Rel0→end)), And(XlteqC(Req14→used_devel,3), XlteqC(Req14→used_testers,2), XlteqC(Req14→used_budget,10000), XeqY(Req14→release_duration,Rel0→duration), XeqY(Req14→release_start,Rel0→start), XeqC(Req14→assigned_release,0)), XneqC(Req14→assigned_release,0))

'Costs for refilling' in Rel0: IfThenElse(And(XgteqY(Req15→start,Rel0→start), XlteqY(Req15→end,Rel0→end)),

And(XlteqC(Req15→used_devel,3), XlteqC(Req15→used_testers,2), XlteqC(Req15→used_budget,10000),
XeqY(Req15→release_duration,Rel0→duration), XeqY(Req15→release_start,Rel0→start), XeqC(Req15→assigned_release,0)),
XneqC(Req15→assigned_release,0))

'Costs for refilling (Store)' in Rel0: IfThenElse(And(XgteqY(Req16→start,Rel0→start), XlteqY(Req16→end,Rel0→end)),
And(XlteqC(Req16→used_devel,3), XlteqC(Req16→used_testers,2), XlteqC(Req16→used_budget,10000),
XeqY(Req16→release_duration,Rel0→duration), XeqY(Req16→release_start,Rel0→start), XeqC(Req16→assigned_release,0)),
XneqC(Req16→assigned_release,0))

'Special Equipment' in Rel0: IfThenElse(And(XgteqY(Req17→start,Rel0→start), XlteqY(Req17→end,Rel0→end)),
And(XlteqC(Req17→used_devel,3), XlteqC(Req17→used_testers,2), XlteqC(Req17→used_budget,10000),
XeqY(Req17→release_duration,Rel0→duration), XeqY(Req17→release_start,Rel0→start), XeqC(Req17→assigned_release,0)),
XneqC(Req17→assigned_release,0))

'Special Equipment (Store)' in Rel0: IfThenElse(And(XgteqY(Req18→start,Rel0→start), XlteqY(Req18→end,Rel0→end)),
And(XlteqC(Req18→used_devel,3), XlteqC(Req18→used_testers,2), XlteqC(Req18→used_budget,10000),
XeqY(Req18→release_duration,Rel0→duration), XeqY(Req18→release_start,Rel0→start), XeqC(Req18→assigned_release,0)),
XneqC(Req18→assigned_release,0))

'Misc. Material' in Rel0: IfThenElse(And(XgteqY(Req19→start,Rel0→start), XlteqY(Req19→end,Rel0→end)),
And(XlteqC(Req19→used_devel,3), XlteqC(Req19→used_testers,2), XlteqC(Req19→used_budget,10000),
XeqY(Req19→release_duration,Rel0→duration), XeqY(Req19→release_start,Rel0→start), XeqC(Req19→assigned_release,0)),
XneqC(Req19→assigned_release,0))

'Misc. Material (Store)' in Rel0: IfThenElse(And(XgteqY(Req20→start,Rel0→start), XlteqY(Req20→end,Rel0→end)),
And(XlteqC(Req20→used_devel,3), XlteqC(Req20→used_testers,2), XlteqC(Req20→used_budget,10000),
XeqY(Req20→release_duration,Rel0→duration), XeqY(Req20→release_start,Rel0→start), XeqC(Req20→assigned_release,0)),
XneqC(Req20→assigned_release,0))

'Misc. Material (GUI)' in Rel0: IfThenElse(And(XgteqY(Req21→start,Rel0→start), XlteqY(Req21→end,Rel0→end)),
And(XlteqC(Req21→used_devel,3), XlteqC(Req21→used_testers,2), XlteqC(Req21→used_budget,10000),
XeqY(Req21→release_duration,Rel0→duration), XeqY(Req21→release_start,Rel0→start), XeqC(Req21→assigned_release,0)),
XneqC(Req21→assigned_release,0))

'Economic efficiency' in Rel0: IfThenElse(And(XgteqY(Req22→start,Rel0→start), XlteqY(Req22→end,Rel0→end)),
And(XlteqC(Req22→used_devel,3), XlteqC(Req22→used_testers,2), XlteqC(Req22→used_budget,10000),
XeqY(Req22→release_duration,Rel0→duration), XeqY(Req22→release_start,Rel0→start), XeqC(Req22→assigned_release,0)),
XneqC(Req22→assigned_release,0))

'Collectors and installation (GUI)' in Rel0: IfThenElse(And(XgteqY(Req23→start,Rel0→start), XlteqY(Req23→end,Rel0→end)),
And(XlteqC(Req23→used_devel,3), XlteqC(Req23→used_testers,2), XlteqC(Req23→used_budget,10000),
XeqY(Req23→release_duration,Rel0→duration), XeqY(Req23→release_start,Rel0→start), XeqC(Req23→assigned_release,0)),
XneqC(Req23→assigned_release,0))

'Pipe costs (GUI-input)' in Rel0: IfThenElse(And(XgteqY(Req24→start,Rel0→start), XlteqY(Req24→end,Rel0→end)),
And(XlteqC(Req24→used_devel,3), XlteqC(Req24→used_testers,2), XlteqC(Req24→used_budget,10000),
XeqY(Req24→release_duration,Rel0→duration), XeqY(Req24→release_start,Rel0→start), XeqC(Req24→assigned_release,0)),
XneqC(Req24→assigned_release,0))

'Air caps calculation (GUI-Input)' in Rel0: IfThenElse(And(XgteqY(Req25→start,Rel0→start), XlteqY(Req25→end,Rel0→end)),
And(XlteqC(Req25→used_devel,3), XlteqC(Req25→used_testers,2), XlteqC(Req25→used_budget,10000),
XeqY(Req25→release_duration,Rel0→duration), XeqY(Req25→release_start,Rel0→start), XeqC(Req25→assigned_release,0)),
XneqC(Req25→assigned_release,0))

'PGR & ADG (GUI-Input)' in Rel0: IfThenElse(And(XgteqY(Req26→start,Rel0→start), XlteqY(Req26→end,Rel0→end)),
And(XlteqC(Req26→used_devel,3), XlteqC(Req26→used_testers,2), XlteqC(Req26→used_budget,10000),
XeqY(Req26→release_duration,Rel0→duration), XeqY(Req26→release_start,Rel0→start), XeqC(Req26→assigned_release,0)),
XneqC(Req26→assigned_release,0))

'Costs for refilling (Calculation)' in Rel0: IfThenElse(And(XgteqY(Req27→start,Rel0→start), XlteqY(Req27→end,Rel0→end)),
And(XlteqC(Req27→used_devel,3), XlteqC(Req27→used_testers,2), XlteqC(Req27→used_budget,10000),
XeqY(Req27→release_duration,Rel0→duration), XeqY(Req27→release_start,Rel0→start), XeqC(Req27→assigned_release,0)),
XneqC(Req27→assigned_release,0))

'Special Equipment (GUI-input)' in Rel0: IfThenElse(And(XgteqY(Req28→start,Rel0→start), XlteqY(Req28→end,Rel0→end)),
And(XlteqC(Req28→used_devel,3), XlteqC(Req28→used_testers,2), XlteqC(Req28→used_budget,10000),
XeqY(Req28→release_duration,Rel0→duration), XeqY(Req28→release_start,Rel0→start), XeqC(Req28→assigned_release,0)),
XneqC(Req28→assigned_release,0))

'PGR & ADG (Calculation) ' in Rel0: IfThenElse(And(XgteqY(Req29→start,Rel0→start), XlteqY(Req29→end,Rel0→end)),
And(XlteqC(Req29→used_devel,3), XlteqC(Req29→used_testers,2), XlteqC(Req29→used_budget,10000),
XeqY(Req29→release_duration,Rel0→duration), XeqY(Req29→release_start,Rel0→start), XeqC(Req29→assigned_release,0)),
XneqC(Req29→assigned_release,0))

'Eco. efficiency (Calc.)' in Rel0: IfThenElse(And(XgteqY(Req30→start,Rel0→start), XlteqY(Req30→end,Rel0→end)),
And(XlteqC(Req30→used_devel,3), XlteqC(Req30→used_testers,2), XlteqC(Req30→used_budget,10000),
XeqY(Req30→release_duration,Rel0→duration), XeqY(Req30→release_start,Rel0→start), XeqC(Req30→assigned_release,0)),

XneqC(Req30→assigned_release,0))

'Eco. efficiency (GUI-input)' in Rel0: IfThenElse(And(XgteqY(Req31→start,Rel0→start), XlteqY(Req31→end,Rel0→end)), And(XlteqC(Req31→used_devel,3), XlteqC(Req31→used_testers,2), XlteqC(Req31→used_budget,10000), XeqY(Req31→release_duration,Rel0→duration), XeqY(Req31→release_start,Rel0→start), XeqC(Req31→assigned_release,0)), XneqC(Req31→assigned_release,0))

'Eco. efficiency (Proposal)' in Rel0: IfThenElse(And(XgteqY(Req32→start,Rel0→start), XlteqY(Req32→end,Rel0→end)), And(XlteqC(Req32→used_devel,3), XlteqC(Req32→used_testers,2), XlteqC(Req32→used_budget,10000), XeqY(Req32→release_duration,Rel0→duration), XeqY(Req32→release_start,Rel0→start), XeqC(Req32→assigned_release,0)), XneqC(Req32→assigned_release,0))

'Modelling an Iconset' in Rel0: IfThenElse(And(XgteqY(Req33→start,Rel0→start), XlteqY(Req33→end,Rel0→end)), And(XlteqC(Req33→used_devel,3), XlteqC(Req33→used_testers,2), XlteqC(Req33→used_budget,10000), XeqY(Req33→release_duration,Rel0→duration), XeqY(Req33→release_start,Rel0→start), XeqC(Req33→assigned_release,0)), XneqC(Req33→assigned_release,0))

'Mysql Backend' in Rel0: IfThenElse(And(XgteqY(Req34→start,Rel0→start), XlteqY(Req34→end,Rel0→end)), And(XlteqC(Req34→used_devel,3), XlteqC(Req34→used_testers,2), XlteqC(Req34→used_budget,10000), XeqY(Req34→release_duration,Rel0→duration), XeqY(Req34→release_start,Rel0→start), XeqC(Req34→assigned_release,0)), XneqC(Req34→assigned_release,0))

'Sqlite Backend' in Rel0: IfThenElse(And(XgteqY(Req35→start,Rel0→start), XlteqY(Req35→end,Rel0→end)), And(XlteqC(Req35→used_devel,3), XlteqC(Req35→used_testers,2), XlteqC(Req35→used_budget,10000), XeqY(Req35→release_duration,Rel0→duration), XeqY(Req35→release_start,Rel0→start), XeqC(Req35→assigned_release,0)), XneqC(Req35→assigned_release,0))

'Costs for refilling (GUI-input)' in Rel0: IfThenElse(And(XgteqY(Req36→start,Rel0→start), XlteqY(Req36→end,Rel0→end)), And(XlteqC(Req36→used_devel,3), XlteqC(Req36→used_testers,2), XlteqC(Req36→used_budget,10000), XeqY(Req36→release_duration,Rel0→duration), XeqY(Req36→release_start,Rel0→start), XeqC(Req36→assigned_release,0)), XneqC(Req36→assigned_release,0))

'Apache Authentication' in Rel0: IfThenElse(And(XgteqY(Req37→start,Rel0→start), XlteqY(Req37→end,Rel0→end)), And(XlteqC(Req37→used_devel,3), XlteqC(Req37→used_testers,2), XlteqC(Req37→used_budget,10000), XeqY(Req37→release_duration,Rel0→duration), XeqY(Req37→release_start,Rel0→start), XeqC(Req37→assigned_release,0)), XneqC(Req37→assigned_release,0))

'DB Auth.' in Rel0: IfThenElse(And(XgteqY(Req38→start,Rel0→start), XlteqY(Req38→end,Rel0→end)), And(XlteqC(Req38→used_devel,3), XlteqC(Req38→used_testers,2), XlteqC(Req38→used_budget,10000), XeqY(Req38→release_duration,Rel0→duration), XeqY(Req38→release_start,Rel0→start), XeqC(Req38→assigned_release,0)), XneqC(Req38→assigned_release,0))

'Auth. Tables' in Rel0: IfThenElse(And(XgteqY(Req39→start,Rel0→start), XlteqY(Req39→end,Rel0→end)), And(XlteqC(Req39→used_devel,3), XlteqC(Req39→used_testers,2), XlteqC(Req39→used_budget,10000), XeqY(Req39→release_duration,Rel0→duration), XeqY(Req39→release_start,Rel0→start), XeqC(Req39→assigned_release,0)), XneqC(Req39→assigned_release,0))

'Auth. DB Communication' in Rel0: IfThenElse(And(XgteqY(Req40→start,Rel0→start), XlteqY(Req40→end,Rel0→end)), And(XlteqC(Req40→used_devel,3), XlteqC(Req40→used_testers,2), XlteqC(Req40→used_budget,10000), XeqY(Req40→release_duration,Rel0→duration), XeqY(Req40→release_start,Rel0→start), XeqC(Req40→assigned_release,0)), XneqC(Req40→assigned_release,0))

'Auth. Pwd Encryption' in Rel0: IfThenElse(And(XgteqY(Req41→start,Rel0→start), XlteqY(Req41→end,Rel0→end)), And(XlteqC(Req41→used_devel,3), XlteqC(Req41→used_testers,2), XlteqC(Req41→used_budget,10000), XeqY(Req41→release_duration,Rel0→duration), XeqY(Req41→release_start,Rel0→start), XeqC(Req41→assigned_release,0)), XneqC(Req41→assigned_release,0))

'Air caps calculation (GUI-Results)' in Rel0: IfThenElse(And(XgteqY(Req42→start,Rel0→start), XlteqY(Req42→end,Rel0→end)), And(XlteqC(Req42→used_devel,3), XlteqC(Req42→used_testers,2), XlteqC(Req42→used_budget,10000), XeqY(Req42→release_duration,Rel0→duration), XeqY(Req42→release_start,Rel0→start), XeqC(Req42→assigned_release,0)), XneqC(Req42→assigned_release,0))

'Pipe costs (GUI-results)' in Rel0: IfThenElse(And(XgteqY(Req43→start,Rel0→start), XlteqY(Req43→end,Rel0→end)), And(XlteqC(Req43→used_devel,3), XlteqC(Req43→used_testers,2), XlteqC(Req43→used_budget,10000), XeqY(Req43→release_duration,Rel0→duration), XeqY(Req43→release_start,Rel0→start), XeqC(Req43→assigned_release,0)), XneqC(Req43→assigned_release,0))

'PGR & ADG (GUI-Result)' in Rel0: IfThenElse(And(XgteqY(Req44→start,Rel0→start), XlteqY(Req44→end,Rel0→end)), And(XlteqC(Req44→used_devel,3), XlteqC(Req44→used_testers,2), XlteqC(Req44→used_budget,10000), XeqY(Req44→release_duration,Rel0→duration), XeqY(Req44→release_start,Rel0→start), XeqC(Req44→assigned_release,0)), XneqC(Req44→assigned_release,0))

'Special Equipment (GUI-results)' in Rel0: IfThenElse(And(XgteqY(Req45→start,Rel0→start), XlteqY(Req45→end,Rel0→end)), And(XlteqC(Req45→used_devel,3), XlteqC(Req45→used_testers,2), XlteqC(Req45→used_budget,10000), XeqY(Req45→release_duration,Rel0→duration), XeqY(Req45→release_start,Rel0→start), XeqC(Req45→assigned_release,0)), XneqC(Req45→assigned_release,0))

'Special Equipment (Calculation)' in Rel0: IfThenElse(And(XgteqY(Req46→start,Rel0→start), XlteqY(Req46→end,Rel0→end)),

And(XlteqC(Req46→used_devel,3), XlteqC(Req46→used_testers,2), XlteqC(Req46→used_budget,10000),
XeqY(Req46→release_duration,Rel0→duration), XeqY(Req46→release_start,Rel0→start), XeqC(Req46→assigned_release,0)),
XneqC(Req46→assigned_release,0))

'Eco. efficiency (Graphics)' in Rel0: IfThenElse(And(XgteqY(Req47→start,Rel0→start), XlteqY(Req47→end,Rel0→end)),
And(XlteqC(Req47→used_devel,3), XlteqC(Req47→used_testers,2), XlteqC(Req47→used_budget,10000),
XeqY(Req47→release_duration,Rel0→duration), XeqY(Req47→release_start,Rel0→start), XeqC(Req47→assigned_release,0)),
XneqC(Req47→assigned_release,0))

'Eco. efficiency (Proposal GUI)' in Rel0: IfThenElse(And(XgteqY(Req48→start,Rel0→start), XlteqY(Req48→end,Rel0→end)),
And(XlteqC(Req48→used_devel,3), XlteqC(Req48→used_testers,2), XlteqC(Req48→used_budget,10000),
XeqY(Req48→release_duration,Rel0→duration), XeqY(Req48→release_start,Rel0→start), XeqC(Req48→assigned_release,0)),
XneqC(Req48→assigned_release,0))

'Multilingualism' in Rel1: IfThenElse(And(XgteqY(Req0→start,Rel1→start), XlteqY(Req0→end,Rel1→end)),
And(XlteqC(Req0→used_devel,3), XlteqC(Req0→used_testers,2), XlteqC(Req0→used_budget,5000),
XeqY(Req0→release_duration,Rel1→duration), XeqY(Req0→release_start,Rel1→start), XeqC(Req0→assigned_release,1)),
XneqC(Req0→assigned_release,1))

'Authentication' in Rel1: IfThenElse(And(XgteqY(Req1→start,Rel1→start), XlteqY(Req1→end,Rel1→end)),
And(XlteqC(Req1→used_devel,3), XlteqC(Req1→used_testers,2), XlteqC(Req1→used_budget,5000),
XeqY(Req1→release_duration,Rel1→duration), XeqY(Req1→release_start,Rel1→start), XeqC(Req1→assigned_release,1)),
XneqC(Req1→assigned_release,1))

'Collectors' surface' in Rel1: IfThenElse(And(XgteqY(Req2→start,Rel1→start), XlteqY(Req2→end,Rel1→end)),
And(XlteqC(Req2→used_devel,3), XlteqC(Req2→used_testers,2), XlteqC(Req2→used_budget,5000),
XeqY(Req2→release_duration,Rel1→duration), XeqY(Req2→release_start,Rel1→start), XeqC(Req2→assigned_release,1)),
XneqC(Req2→assigned_release,1))

'GUI Preconditions' in Rel1: IfThenElse(And(XgteqY(Req3→start,Rel1→start), XlteqY(Req3→end,Rel1→end)),
And(XlteqC(Req3→used_devel,3), XlteqC(Req3→used_testers,2), XlteqC(Req3→used_budget,5000),
XeqY(Req3→release_duration,Rel1→duration), XeqY(Req3→release_start,Rel1→start), XeqC(Req3→assigned_release,1)),
XneqC(Req3→assigned_release,1))

'Collectors' surface (GUI)' in Rel1: IfThenElse(And(XgteqY(Req4→start,Rel1→start), XlteqY(Req4→end,Rel1→end)),
And(XlteqC(Req4→used_devel,3), XlteqC(Req4→used_testers,2), XlteqC(Req4→used_budget,5000),
XeqY(Req4→release_duration,Rel1→duration), XeqY(Req4→release_start,Rel1→start), XeqC(Req4→assigned_release,1)),
XneqC(Req4→assigned_release,1))

'Database Backend' in Rel1: IfThenElse(And(XgteqY(Req5→start,Rel1→start), XlteqY(Req5→end,Rel1→end)),
And(XlteqC(Req5→used_devel,3), XlteqC(Req5→used_testers,2), XlteqC(Req5→used_budget,5000),
XeqY(Req5→release_duration,Rel1→duration), XeqY(Req5→release_start,Rel1→start), XeqC(Req5→assigned_release,1)),
XneqC(Req5→assigned_release,1))

'Collectors and installation' in Rel1: IfThenElse(And(XgteqY(Req6→start,Rel1→start), XlteqY(Req6→end,Rel1→end)),
And(XlteqC(Req6→used_devel,3), XlteqC(Req6→used_testers,2), XlteqC(Req6→used_budget,5000),
XeqY(Req6→release_duration,Rel1→duration), XeqY(Req6→release_start,Rel1→start), XeqC(Req6→assigned_release,1)),
XneqC(Req6→assigned_release,1))

'Collectors's surface (Store)' in Rel1: IfThenElse(And(XgteqY(Req7→start,Rel1→start), XlteqY(Req7→end,Rel1→end)),
And(XlteqC(Req7→used_devel,3), XlteqC(Req7→used_testers,2), XlteqC(Req7→used_budget,5000),
XeqY(Req7→release_duration,Rel1→duration), XeqY(Req7→release_start,Rel1→start), XeqC(Req7→assigned_release,1)),
XneqC(Req7→assigned_release,1))

'Collectors and installation (Store)' in Rel1: IfThenElse(And(XgteqY(Req8→start,Rel1→start), XlteqY(Req8→end,Rel1→end)),
And(XlteqC(Req8→used_devel,3), XlteqC(Req8→used_testers,2), XlteqC(Req8→used_budget,5000),
XeqY(Req8→release_duration,Rel1→duration), XeqY(Req8→release_start,Rel1→start), XeqC(Req8→assigned_release,1)),
XneqC(Req8→assigned_release,1))

'Air caps calculation' in Rel1: IfThenElse(And(XgteqY(Req9→start,Rel1→start), XlteqY(Req9→end,Rel1→end)),
And(XlteqC(Req9→used_devel,3), XlteqC(Req9→used_testers,2), XlteqC(Req9→used_budget,5000),
XeqY(Req9→release_duration,Rel1→duration), XeqY(Req9→release_start,Rel1→start), XeqC(Req9→assigned_release,1)),
XneqC(Req9→assigned_release,1))

'Air caps calculation (Store)' in Rel1: IfThenElse(And(XgteqY(Req10→start,Rel1→start), XlteqY(Req10→end,Rel1→end)),
And(XlteqC(Req10→used_devel,3), XlteqC(Req10→used_testers,2), XlteqC(Req10→used_budget,5000),
XeqY(Req10→release_duration,Rel1→duration), XeqY(Req10→release_start,Rel1→start), XeqC(Req10→assigned_release,1)),
XneqC(Req10→assigned_release,1))

'Pipe Costs' in Rel1: IfThenElse(And(XgteqY(Req11→start,Rel1→start), XlteqY(Req11→end,Rel1→end)),
And(XlteqC(Req11→used_devel,3), XlteqC(Req11→used_testers,2), XlteqC(Req11→used_budget,5000),
XeqY(Req11→release_duration,Rel1→duration), XeqY(Req11→release_start,Rel1→start), XeqC(Req11→assigned_release,1)),
XneqC(Req11→assigned_release,1))

'Pipe Costs (Store)' in Rel1: IfThenElse(And(XgteqY(Req12→start,Rel1→start), XlteqY(Req12→end,Rel1→end)),
And(XlteqC(Req12→used_devel,3), XlteqC(Req12→used_testers,2), XlteqC(Req12→used_budget,5000),
XeqY(Req12→release_duration,Rel1→duration), XeqY(Req12→release_start,Rel1→start), XeqC(Req12→assigned_release,1)),

XneqC(Req12→assigned_release,1))

'PGR & ADG' in Rel1: IfThenElse(And(XgteqY(Req13→start,Rel1→start), XlteqY(Req13→end,Rel1→end)), And(XlteqC(Req13→used_devel,3), XlteqC(Req13→used_testers,2), XlteqC(Req13→used_budget,5000), XeqY(Req13→release_duration,Rel1→duration), XeqY(Req13→release_start,Rel1→start), XeqC(Req13→assigned_release,1)), XneqC(Req13→assigned_release,1))

'PGR & ADG (Store)' in Rel1: IfThenElse(And(XgteqY(Req14→start,Rel1→start), XlteqY(Req14→end,Rel1→end)), And(XlteqC(Req14→used_devel,3), XlteqC(Req14→used_testers,2), XlteqC(Req14→used_budget,5000), XeqY(Req14→release_duration,Rel1→duration), XeqY(Req14→release_start,Rel1→start), XeqC(Req14→assigned_release,1)), XneqC(Req14→assigned_release,1))

'Costs for refilling' in Rel1: IfThenElse(And(XgteqY(Req15→start,Rel1→start), XlteqY(Req15→end,Rel1→end)), And(XlteqC(Req15→used_devel,3), XlteqC(Req15→used_testers,2), XlteqC(Req15→used_budget,5000), XeqY(Req15→release_duration,Rel1→duration), XeqY(Req15→release_start,Rel1→start), XeqC(Req15→assigned_release,1)), XneqC(Req15→assigned_release,1))

'Costs for refilling (Store)' in Rel1: IfThenElse(And(XgteqY(Req16→start,Rel1→start), XlteqY(Req16→end,Rel1→end)), And(XlteqC(Req16→used_devel,3), XlteqC(Req16→used_testers,2), XlteqC(Req16→used_budget,5000), XeqY(Req16→release_duration,Rel1→duration), XeqY(Req16→release_start,Rel1→start), XeqC(Req16→assigned_release,1)), XneqC(Req16→assigned_release,1))

'Special Equipment' in Rel1: IfThenElse(And(XgteqY(Req17→start,Rel1→start), XlteqY(Req17→end,Rel1→end)), And(XlteqC(Req17→used_devel,3), XlteqC(Req17→used_testers,2), XlteqC(Req17→used_budget,5000), XeqY(Req17→release_duration,Rel1→duration), XeqY(Req17→release_start,Rel1→start), XeqC(Req17→assigned_release,1)), XneqC(Req17→assigned_release,1))

'Special Equipment (Store)' in Rel1: IfThenElse(And(XgteqY(Req18→start,Rel1→start), XlteqY(Req18→end,Rel1→end)), And(XlteqC(Req18→used_devel,3), XlteqC(Req18→used_testers,2), XlteqC(Req18→used_budget,5000), XeqY(Req18→release_duration,Rel1→duration), XeqY(Req18→release_start,Rel1→start), XeqC(Req18→assigned_release,1)), XneqC(Req18→assigned_release,1))

'Misc. Material' in Rel1: IfThenElse(And(XgteqY(Req19→start,Rel1→start), XlteqY(Req19→end,Rel1→end)), And(XlteqC(Req19→used_devel,3), XlteqC(Req19→used_testers,2), XlteqC(Req19→used_budget,5000), XeqY(Req19→release_duration,Rel1→duration), XeqY(Req19→release_start,Rel1→start), XeqC(Req19→assigned_release,1)), XneqC(Req19→assigned_release,1))

'Misc. Material (Store)' in Rel1: IfThenElse(And(XgteqY(Req20→start,Rel1→start), XlteqY(Req20→end,Rel1→end)), And(XlteqC(Req20→used_devel,3), XlteqC(Req20→used_testers,2), XlteqC(Req20→used_budget,5000), XeqY(Req20→release_duration,Rel1→duration), XeqY(Req20→release_start,Rel1→start), XeqC(Req20→assigned_release,1)), XneqC(Req20→assigned_release,1))

'Misc. Material (GUI)' in Rel1: IfThenElse(And(XgteqY(Req21→start,Rel1→start), XlteqY(Req21→end,Rel1→end)), And(XlteqC(Req21→used_devel,3), XlteqC(Req21→used_testers,2), XlteqC(Req21→used_budget,5000), XeqY(Req21→release_duration,Rel1→duration), XeqY(Req21→release_start,Rel1→start), XeqC(Req21→assigned_release,1)), XneqC(Req21→assigned_release,1))

'Economic efficiency' in Rel1: IfThenElse(And(XgteqY(Req22→start,Rel1→start), XlteqY(Req22→end,Rel1→end)), And(XlteqC(Req22→used_devel,3), XlteqC(Req22→used_testers,2), XlteqC(Req22→used_budget,5000), XeqY(Req22→release_duration,Rel1→duration), XeqY(Req22→release_start,Rel1→start), XeqC(Req22→assigned_release,1)), XneqC(Req22→assigned_release,1))

'Collectors and installation (GUI)' in Rel1: IfThenElse(And(XgteqY(Req23→start,Rel1→start), XlteqY(Req23→end,Rel1→end)), And(XlteqC(Req23→used_devel,3), XlteqC(Req23→used_testers,2), XlteqC(Req23→used_budget,5000), XeqY(Req23→release_duration,Rel1→duration), XeqY(Req23→release_start,Rel1→start), XeqC(Req23→assigned_release,1)), XneqC(Req23→assigned_release,1))

'Pipe costs (GUI-input)' in Rel1: IfThenElse(And(XgteqY(Req24→start,Rel1→start), XlteqY(Req24→end,Rel1→end)), And(XlteqC(Req24→used_devel,3), XlteqC(Req24→used_testers,2), XlteqC(Req24→used_budget,5000), XeqY(Req24→release_duration,Rel1→duration), XeqY(Req24→release_start,Rel1→start), XeqC(Req24→assigned_release,1)), XneqC(Req24→assigned_release,1))

'Air caps calculation (GUI-Input)' in Rel1: IfThenElse(And(XgteqY(Req25→start,Rel1→start), XlteqY(Req25→end,Rel1→end)), And(XlteqC(Req25→used_devel,3), XlteqC(Req25→used_testers,2), XlteqC(Req25→used_budget,5000), XeqY(Req25→release_duration,Rel1→duration), XeqY(Req25→release_start,Rel1→start), XeqC(Req25→assigned_release,1)), XneqC(Req25→assigned_release,1))

'PGR & ADG (GUI-Input)' in Rel1: IfThenElse(And(XgteqY(Req26→start,Rel1→start), XlteqY(Req26→end,Rel1→end)), And(XlteqC(Req26→used_devel,3), XlteqC(Req26→used_testers,2), XlteqC(Req26→used_budget,5000), XeqY(Req26→release_duration,Rel1→duration), XeqY(Req26→release_start,Rel1→start), XeqC(Req26→assigned_release,1)), XneqC(Req26→assigned_release,1))

'Costs for refilling (Calculation)' in Rel1: IfThenElse(And(XgteqY(Req27→start,Rel1→start), XlteqY(Req27→end,Rel1→end)), And(XlteqC(Req27→used_devel,3), XlteqC(Req27→used_testers,2), XlteqC(Req27→used_budget,5000), XeqY(Req27→release_duration,Rel1→duration), XeqY(Req27→release_start,Rel1→start), XeqC(Req27→assigned_release,1)), XneqC(Req27→assigned_release,1))

'Special Equipment (GUI-input)' in Rel1: IfThenElse(And(XgteqY(Req28→start,Rel1→start), XlteqY(Req28→end,Rel1→end)),

And(XlteqC(Req28→used_devel,3), XlteqC(Req28→used_testers,2), XlteqC(Req28→used_budget,5000),
XeqY(Req28→release_duration,Rel1→duration), XeqY(Req28→release_start,Rel1→start), XeqC(Req28→assigned_release,1)),
XneqC(Req28→assigned_release,1))

'PGR & ADG (Calculation) ' in Rel1: IfThenElse(And(XgteqY(Req29→start,Rel1→start), XlteqY(Req29→end,Rel1→end)),
And(XlteqC(Req29→used_devel,3), XlteqC(Req29→used_testers,2), XlteqC(Req29→used_budget,5000),
XeqY(Req29→release_duration,Rel1→duration), XeqY(Req29→release_start,Rel1→start), XeqC(Req29→assigned_release,1)),
XneqC(Req29→assigned_release,1))

'Eco. efficiency (Calc.)' in Rel1: IfThenElse(And(XgteqY(Req30→start,Rel1→start), XlteqY(Req30→end,Rel1→end)),
And(XlteqC(Req30→used_devel,3), XlteqC(Req30→used_testers,2), XlteqC(Req30→used_budget,5000),
XeqY(Req30→release_duration,Rel1→duration), XeqY(Req30→release_start,Rel1→start), XeqC(Req30→assigned_release,1)),
XneqC(Req30→assigned_release,1))

'Eco. efficiency (GUI-input)' in Rel1: IfThenElse(And(XgteqY(Req31→start,Rel1→start), XlteqY(Req31→end,Rel1→end)),
And(XlteqC(Req31→used_devel,3), XlteqC(Req31→used_testers,2), XlteqC(Req31→used_budget,5000),
XeqY(Req31→release_duration,Rel1→duration), XeqY(Req31→release_start,Rel1→start), XeqC(Req31→assigned_release,1)),
XneqC(Req31→assigned_release,1))

'Eco. efficiency (Proposal)' in Rel1: IfThenElse(And(XgteqY(Req32→start,Rel1→start), XlteqY(Req32→end,Rel1→end)),
And(XlteqC(Req32→used_devel,3), XlteqC(Req32→used_testers,2), XlteqC(Req32→used_budget,5000),
XeqY(Req32→release_duration,Rel1→duration), XeqY(Req32→release_start,Rel1→start), XeqC(Req32→assigned_release,1)),
XneqC(Req32→assigned_release,1))

'Modelling an Iconset' in Rel1: IfThenElse(And(XgteqY(Req33→start,Rel1→start), XlteqY(Req33→end,Rel1→end)),
And(XlteqC(Req33→used_devel,3), XlteqC(Req33→used_testers,2), XlteqC(Req33→used_budget,5000),
XeqY(Req33→release_duration,Rel1→duration), XeqY(Req33→release_start,Rel1→start), XeqC(Req33→assigned_release,1)),
XneqC(Req33→assigned_release,1))

'Mysql Backend' in Rel1: IfThenElse(And(XgteqY(Req34→start,Rel1→start), XlteqY(Req34→end,Rel1→end)),
And(XlteqC(Req34→used_devel,3), XlteqC(Req34→used_testers,2), XlteqC(Req34→used_budget,5000),
XeqY(Req34→release_duration,Rel1→duration), XeqY(Req34→release_start,Rel1→start), XeqC(Req34→assigned_release,1)),
XneqC(Req34→assigned_release,1))

'Sqlite Backend' in Rel1: IfThenElse(And(XgteqY(Req35→start,Rel1→start), XlteqY(Req35→end,Rel1→end)),
And(XlteqC(Req35→used_devel,3), XlteqC(Req35→used_testers,2), XlteqC(Req35→used_budget,5000),
XeqY(Req35→release_duration,Rel1→duration), XeqY(Req35→release_start,Rel1→start), XeqC(Req35→assigned_release,1)),
XneqC(Req35→assigned_release,1))

'Costs for refilling (GUI-input)' in Rel1: IfThenElse(And(XgteqY(Req36→start,Rel1→start), XlteqY(Req36→end,Rel1→end)),
And(XlteqC(Req36→used_devel,3), XlteqC(Req36→used_testers,2), XlteqC(Req36→used_budget,5000),
XeqY(Req36→release_duration,Rel1→duration), XeqY(Req36→release_start,Rel1→start), XeqC(Req36→assigned_release,1)),
XneqC(Req36→assigned_release,1))

'Apache Authentication' in Rel1: IfThenElse(And(XgteqY(Req37→start,Rel1→start), XlteqY(Req37→end,Rel1→end)),
And(XlteqC(Req37→used_devel,3), XlteqC(Req37→used_testers,2), XlteqC(Req37→used_budget,5000),
XeqY(Req37→release_duration,Rel1→duration), XeqY(Req37→release_start,Rel1→start), XeqC(Req37→assigned_release,1)),
XneqC(Req37→assigned_release,1))

'DB Auth.' in Rel1: IfThenElse(And(XgteqY(Req38→start,Rel1→start), XlteqY(Req38→end,Rel1→end)),
And(XlteqC(Req38→used_devel,3), XlteqC(Req38→used_testers,2), XlteqC(Req38→used_budget,5000),
XeqY(Req38→release_duration,Rel1→duration), XeqY(Req38→release_start,Rel1→start), XeqC(Req38→assigned_release,1)),
XneqC(Req38→assigned_release,1))

'Auth. Tables' in Rel1: IfThenElse(And(XgteqY(Req39→start,Rel1→start), XlteqY(Req39→end,Rel1→end)),
And(XlteqC(Req39→used_devel,3), XlteqC(Req39→used_testers,2), XlteqC(Req39→used_budget,5000),
XeqY(Req39→release_duration,Rel1→duration), XeqY(Req39→release_start,Rel1→start), XeqC(Req39→assigned_release,1)),
XneqC(Req39→assigned_release,1))

'Auth. DB Communication' in Rel1: IfThenElse(And(XgteqY(Req40→start,Rel1→start), XlteqY(Req40→end,Rel1→end)),
And(XlteqC(Req40→used_devel,3), XlteqC(Req40→used_testers,2), XlteqC(Req40→used_budget,5000),
XeqY(Req40→release_duration,Rel1→duration), XeqY(Req40→release_start,Rel1→start), XeqC(Req40→assigned_release,1)),
XneqC(Req40→assigned_release,1))

'Auth. Pwd Encryption' in Rel1: IfThenElse(And(XgteqY(Req41→start,Rel1→start), XlteqY(Req41→end,Rel1→end)),
And(XlteqC(Req41→used_devel,3), XlteqC(Req41→used_testers,2), XlteqC(Req41→used_budget,5000),
XeqY(Req41→release_duration,Rel1→duration), XeqY(Req41→release_start,Rel1→start), XeqC(Req41→assigned_release,1)),
XneqC(Req41→assigned_release,1))

'Air caps calculation (GUI-Results)' in Rel1: IfThenElse(And(XgteqY(Req42→start,Rel1→start), XlteqY(Req42→end,Rel1→end)),
And(XlteqC(Req42→used_devel,3), XlteqC(Req42→used_testers,2), XlteqC(Req42→used_budget,5000),
XeqY(Req42→release_duration,Rel1→duration), XeqY(Req42→release_start,Rel1→start), XeqC(Req42→assigned_release,1)),
XneqC(Req42→assigned_release,1))

'Pipe costs (GUI-results)' in Rel1: IfThenElse(And(XgteqY(Req43→start,Rel1→start), XlteqY(Req43→end,Rel1→end)),
And(XlteqC(Req43→used_devel,3), XlteqC(Req43→used_testers,2), XlteqC(Req43→used_budget,5000),
XeqY(Req43→release_duration,Rel1→duration), XeqY(Req43→release_start,Rel1→start), XeqC(Req43→assigned_release,1)),

XneqC(Req43→assigned_release,1))

'PGR & ADG (GUI-Result)' in Rel1: IfThenElse(And(XgteqY(Req44→start,Rel1→start), XlteqY(Req44→end,Rel1→end)), And(XlteqC(Req44→used_devel,3), XlteqC(Req44→used_testers,2), XlteqC(Req44→used_budget,5000), XeqY(Req44→release_duration,Rel1→duration), XeqY(Req44→release_start,Rel1→start), XeqC(Req44→assigned_release,1)), XneqC(Req44→assigned_release,1))

'Special Equipment (GUI-results)' in Rel1: IfThenElse(And(XgteqY(Req45→start,Rel1→start), XlteqY(Req45→end,Rel1→end)), And(XlteqC(Req45→used_devel,3), XlteqC(Req45→used_testers,2), XlteqC(Req45→used_budget,5000), XeqY(Req45→release_duration,Rel1→duration), XeqY(Req45→release_start,Rel1→start), XeqC(Req45→assigned_release,1)), XneqC(Req45→assigned_release,1))

'Special Equipment (Calculation)' in Rel1: IfThenElse(And(XgteqY(Req46→start,Rel1→start), XlteqY(Req46→end,Rel1→end)), And(XlteqC(Req46→used_devel,3), XlteqC(Req46→used_testers,2), XlteqC(Req46→used_budget,5000), XeqY(Req46→release_duration,Rel1→duration), XeqY(Req46→release_start,Rel1→start), XeqC(Req46→assigned_release,1)), XneqC(Req46→assigned_release,1))

'Eco. efficiency (Graphics)' in Rel1: IfThenElse(And(XgteqY(Req47→start,Rel1→start), XlteqY(Req47→end,Rel1→end)), And(XlteqC(Req47→used_devel,3), XlteqC(Req47→used_testers,2), XlteqC(Req47→used_budget,5000), XeqY(Req47→release_duration,Rel1→duration), XeqY(Req47→release_start,Rel1→start), XeqC(Req47→assigned_release,1)), XneqC(Req47→assigned_release,1))

'Eco. efficiency (Proposal GUI)' in Rel1: IfThenElse(And(XgteqY(Req48→start,Rel1→start), XlteqY(Req48→end,Rel1→end)), And(XlteqC(Req48→used_devel,3), XlteqC(Req48→used_testers,2), XlteqC(Req48→used_budget,5000), XeqY(Req48→release_duration,Rel1→duration), XeqY(Req48→release_start,Rel1→start), XeqC(Req48→assigned_release,1)), XneqC(Req48→assigned_release,1))

'Multilingualism' in Rel2: IfThenElse(And(XgteqY(Req0→start,Rel2→start), XlteqY(Req0→end,Rel2→end)), And(XlteqC(Req0→used_devel,2), XlteqC(Req0→used_testers,1), XlteqC(Req0→used_budget,5000), XeqY(Req0→release_duration,Rel2→duration), XeqY(Req0→release_start,Rel2→start), XeqC(Req0→assigned_release,2)), XneqC(Req0→assigned_release,2))

'Authentication' in Rel2: IfThenElse(And(XgteqY(Req1→start,Rel2→start), XlteqY(Req1→end,Rel2→end)), And(XlteqC(Req1→used_devel,2), XlteqC(Req1→used_testers,1), XlteqC(Req1→used_budget,5000), XeqY(Req1→release_duration,Rel2→duration), XeqY(Req1→release_start,Rel2→start), XeqC(Req1→assigned_release,2)), XneqC(Req1→assigned_release,2))

'Collectors' surface' in Rel2: IfThenElse(And(XgteqY(Req2→start,Rel2→start), XlteqY(Req2→end,Rel2→end)), And(XlteqC(Req2→used_devel,2), XlteqC(Req2→used_testers,1), XlteqC(Req2→used_budget,5000), XeqY(Req2→release_duration,Rel2→duration), XeqY(Req2→release_start,Rel2→start), XeqC(Req2→assigned_release,2)), XneqC(Req2→assigned_release,2))

'GUI Preconditions' in Rel2: IfThenElse(And(XgteqY(Req3→start,Rel2→start), XlteqY(Req3→end,Rel2→end)), And(XlteqC(Req3→used_devel,2), XlteqC(Req3→used_testers,1), XlteqC(Req3→used_budget,5000), XeqY(Req3→release_duration,Rel2→duration), XeqY(Req3→release_start,Rel2→start), XeqC(Req3→assigned_release,2)), XneqC(Req3→assigned_release,2))

'Collectors' surface (GUI)' in Rel2: IfThenElse(And(XgteqY(Req4→start,Rel2→start), XlteqY(Req4→end,Rel2→end)), And(XlteqC(Req4→used_devel,2), XlteqC(Req4→used_testers,1), XlteqC(Req4→used_budget,5000), XeqY(Req4→release_duration,Rel2→duration), XeqY(Req4→release_start,Rel2→start), XeqC(Req4→assigned_release,2)), XneqC(Req4→assigned_release,2))

'Database Backend' in Rel2: IfThenElse(And(XgteqY(Req5→start,Rel2→start), XlteqY(Req5→end,Rel2→end)), And(XlteqC(Req5→used_devel,2), XlteqC(Req5→used_testers,1), XlteqC(Req5→used_budget,5000), XeqY(Req5→release_duration,Rel2→duration), XeqY(Req5→release_start,Rel2→start), XeqC(Req5→assigned_release,2)), XneqC(Req5→assigned_release,2))

'Collectors and installation' in Rel2: IfThenElse(And(XgteqY(Req6→start,Rel2→start), XlteqY(Req6→end,Rel2→end)), And(XlteqC(Req6→used_devel,2), XlteqC(Req6→used_testers,1), XlteqC(Req6→used_budget,5000), XeqY(Req6→release_duration,Rel2→duration), XeqY(Req6→release_start,Rel2→start), XeqC(Req6→assigned_release,2)), XneqC(Req6→assigned_release,2))

'Collectors's surface (Store)' in Rel2: IfThenElse(And(XgteqY(Req7→start,Rel2→start), XlteqY(Req7→end,Rel2→end)), And(XlteqC(Req7→used_devel,2), XlteqC(Req7→used_testers,1), XlteqC(Req7→used_budget,5000), XeqY(Req7→release_duration,Rel2→duration), XeqY(Req7→release_start,Rel2→start), XeqC(Req7→assigned_release,2)), XneqC(Req7→assigned_release,2))

'Collectors and installation (Store)' in Rel2: IfThenElse(And(XgteqY(Req8→start,Rel2→start), XlteqY(Req8→end,Rel2→end)), And(XlteqC(Req8→used_devel,2), XlteqC(Req8→used_testers,1), XlteqC(Req8→used_budget,5000), XeqY(Req8→release_duration,Rel2→duration), XeqY(Req8→release_start,Rel2→start), XeqC(Req8→assigned_release,2)), XneqC(Req8→assigned_release,2))

'Air caps calculation' in Rel2: IfThenElse(And(XgteqY(Req9→start,Rel2→start), XlteqY(Req9→end,Rel2→end)), And(XlteqC(Req9→used_devel,2), XlteqC(Req9→used_testers,1), XlteqC(Req9→used_budget,5000), XeqY(Req9→release_duration,Rel2→duration), XeqY(Req9→release_start,Rel2→start), XeqC(Req9→assigned_release,2)), XneqC(Req9→assigned_release,2))

'Air caps calculation (Store)' in Rel2: IfThenElse(And(XgteqY(Req10→start,Rel2→start), XlteqY(Req10→end,Rel2→end)),

And(XlteqC(Req10→used_devel,2), XlteqC(Req10→used_testers,1), XlteqC(Req10→used_budget,5000),
XeqY(Req10→release_duration,Rel2→duration), XeqY(Req10→release_start,Rel2→start), XeqC(Req10→assigned_release,2)),
XneqC(Req10→assigned_release,2))

'Pipe Costs' in Rel2: IfThenElse(And(XgteqY(Req11→start,Rel2→start), XlteqY(Req11→end,Rel2→end)),
And(XlteqC(Req11→used_devel,2), XlteqC(Req11→used_testers,1), XlteqC(Req11→used_budget,5000),
XeqY(Req11→release_duration,Rel2→duration), XeqY(Req11→release_start,Rel2→start), XeqC(Req11→assigned_release,2)),
XneqC(Req11→assigned_release,2))

'Pipe Costs (Store)' in Rel2: IfThenElse(And(XgteqY(Req12→start,Rel2→start), XlteqY(Req12→end,Rel2→end)),
And(XlteqC(Req12→used_devel,2), XlteqC(Req12→used_testers,1), XlteqC(Req12→used_budget,5000),
XeqY(Req12→release_duration,Rel2→duration), XeqY(Req12→release_start,Rel2→start), XeqC(Req12→assigned_release,2)),
XneqC(Req12→assigned_release,2))

'PGR & ADG' in Rel2: IfThenElse(And(XgteqY(Req13→start,Rel2→start), XlteqY(Req13→end,Rel2→end)),
And(XlteqC(Req13→used_devel,2), XlteqC(Req13→used_testers,1), XlteqC(Req13→used_budget,5000),
XeqY(Req13→release_duration,Rel2→duration), XeqY(Req13→release_start,Rel2→start), XeqC(Req13→assigned_release,2)),
XneqC(Req13→assigned_release,2))

'PGR & ADG (Store)' in Rel2: IfThenElse(And(XgteqY(Req14→start,Rel2→start), XlteqY(Req14→end,Rel2→end)),
And(XlteqC(Req14→used_devel,2), XlteqC(Req14→used_testers,1), XlteqC(Req14→used_budget,5000),
XeqY(Req14→release_duration,Rel2→duration), XeqY(Req14→release_start,Rel2→start), XeqC(Req14→assigned_release,2)),
XneqC(Req14→assigned_release,2))

'Costs for refilling' in Rel2: IfThenElse(And(XgteqY(Req15→start,Rel2→start), XlteqY(Req15→end,Rel2→end)),
And(XlteqC(Req15→used_devel,2), XlteqC(Req15→used_testers,1), XlteqC(Req15→used_budget,5000),
XeqY(Req15→release_duration,Rel2→duration), XeqY(Req15→release_start,Rel2→start), XeqC(Req15→assigned_release,2)),
XneqC(Req15→assigned_release,2))

'Costs for refilling (Store)' in Rel2: IfThenElse(And(XgteqY(Req16→start,Rel2→start), XlteqY(Req16→end,Rel2→end)),
And(XlteqC(Req16→used_devel,2), XlteqC(Req16→used_testers,1), XlteqC(Req16→used_budget,5000),
XeqY(Req16→release_duration,Rel2→duration), XeqY(Req16→release_start,Rel2→start), XeqC(Req16→assigned_release,2)),
XneqC(Req16→assigned_release,2))

'Special Equipment' in Rel2: IfThenElse(And(XgteqY(Req17→start,Rel2→start), XlteqY(Req17→end,Rel2→end)),
And(XlteqC(Req17→used_devel,2), XlteqC(Req17→used_testers,1), XlteqC(Req17→used_budget,5000),
XeqY(Req17→release_duration,Rel2→duration), XeqY(Req17→release_start,Rel2→start), XeqC(Req17→assigned_release,2)),
XneqC(Req17→assigned_release,2))

'Special Equipment (Store)' in Rel2: IfThenElse(And(XgteqY(Req18→start,Rel2→start), XlteqY(Req18→end,Rel2→end)),
And(XlteqC(Req18→used_devel,2), XlteqC(Req18→used_testers,1), XlteqC(Req18→used_budget,5000),
XeqY(Req18→release_duration,Rel2→duration), XeqY(Req18→release_start,Rel2→start), XeqC(Req18→assigned_release,2)),
XneqC(Req18→assigned_release,2))

'Misc. Material' in Rel2: IfThenElse(And(XgteqY(Req19→start,Rel2→start), XlteqY(Req19→end,Rel2→end)),
And(XlteqC(Req19→used_devel,2), XlteqC(Req19→used_testers,1), XlteqC(Req19→used_budget,5000),
XeqY(Req19→release_duration,Rel2→duration), XeqY(Req19→release_start,Rel2→start), XeqC(Req19→assigned_release,2)),
XneqC(Req19→assigned_release,2))

'Misc. Material (Store)' in Rel2: IfThenElse(And(XgteqY(Req20→start,Rel2→start), XlteqY(Req20→end,Rel2→end)),
And(XlteqC(Req20→used_devel,2), XlteqC(Req20→used_testers,1), XlteqC(Req20→used_budget,5000),
XeqY(Req20→release_duration,Rel2→duration), XeqY(Req20→release_start,Rel2→start), XeqC(Req20→assigned_release,2)),
XneqC(Req20→assigned_release,2))

'Misc. Material (GUI)' in Rel2: IfThenElse(And(XgteqY(Req21→start,Rel2→start), XlteqY(Req21→end,Rel2→end)),
And(XlteqC(Req21→used_devel,2), XlteqC(Req21→used_testers,1), XlteqC(Req21→used_budget,5000),
XeqY(Req21→release_duration,Rel2→duration), XeqY(Req21→release_start,Rel2→start), XeqC(Req21→assigned_release,2)),
XneqC(Req21→assigned_release,2))

'Economic efficiency' in Rel2: IfThenElse(And(XgteqY(Req22→start,Rel2→start), XlteqY(Req22→end,Rel2→end)),
And(XlteqC(Req22→used_devel,2), XlteqC(Req22→used_testers,1), XlteqC(Req22→used_budget,5000),
XeqY(Req22→release_duration,Rel2→duration), XeqY(Req22→release_start,Rel2→start), XeqC(Req22→assigned_release,2)),
XneqC(Req22→assigned_release,2))

'Collectors and installation (GUI)' in Rel2: IfThenElse(And(XgteqY(Req23→start,Rel2→start), XlteqY(Req23→end,Rel2→end)),
And(XlteqC(Req23→used_devel,2), XlteqC(Req23→used_testers,1), XlteqC(Req23→used_budget,5000),
XeqY(Req23→release_duration,Rel2→duration), XeqY(Req23→release_start,Rel2→start), XeqC(Req23→assigned_release,2)),
XneqC(Req23→assigned_release,2))

'Pipe costs (GUI-input)' in Rel2: IfThenElse(And(XgteqY(Req24→start,Rel2→start), XlteqY(Req24→end,Rel2→end)),
And(XlteqC(Req24→used_devel,2), XlteqC(Req24→used_testers,1), XlteqC(Req24→used_budget,5000),
XeqY(Req24→release_duration,Rel2→duration), XeqY(Req24→release_start,Rel2→start), XeqC(Req24→assigned_release,2)),
XneqC(Req24→assigned_release,2))

'Air caps calculation (GUI-Input)' in Rel2: IfThenElse(And(XgteqY(Req25→start,Rel2→start), XlteqY(Req25→end,Rel2→end)),
And(XlteqC(Req25→used_devel,2), XlteqC(Req25→used_testers,1), XlteqC(Req25→used_budget,5000),
XeqY(Req25→release_duration,Rel2→duration), XeqY(Req25→release_start,Rel2→start), XeqC(Req25→assigned_release,2)),

XneqC(Req25→assigned_release,2))

'PGR & ADG (GUI-Input)' in Rel2: IfThenElse(And(XgteqY(Req26→start,Rel2→start), XlteqY(Req26→end,Rel2→end)), And(XlteqC(Req26→used_devel,2), XlteqC(Req26→used_testers,1), XlteqC(Req26→used_budget,5000), XeqY(Req26→release_duration,Rel2→duration), XeqY(Req26→release_start,Rel2→start), XeqC(Req26→assigned_release,2)), XneqC(Req26→assigned_release,2))

'Costs for refilling (Calculation)' in Rel2: IfThenElse(And(XgteqY(Req27→start,Rel2→start), XlteqY(Req27→end,Rel2→end)), And(XlteqC(Req27→used_devel,2), XlteqC(Req27→used_testers,1), XlteqC(Req27→used_budget,5000), XeqY(Req27→release_duration,Rel2→duration), XeqY(Req27→release_start,Rel2→start), XeqC(Req27→assigned_release,2)), XneqC(Req27→assigned_release,2))

'Special Equipment (GUI-input)' in Rel2: IfThenElse(And(XgteqY(Req28→start,Rel2→start), XlteqY(Req28→end,Rel2→end)), And(XlteqC(Req28→used_devel,2), XlteqC(Req28→used_testers,1), XlteqC(Req28→used_budget,5000), XeqY(Req28→release_duration,Rel2→duration), XeqY(Req28→release_start,Rel2→start), XeqC(Req28→assigned_release,2)), XneqC(Req28→assigned_release,2))

'PGR & ADG (Calculation) ' in Rel2: IfThenElse(And(XgteqY(Req29→start,Rel2→start), XlteqY(Req29→end,Rel2→end)), And(XlteqC(Req29→used_devel,2), XlteqC(Req29→used_testers,1), XlteqC(Req29→used_budget,5000), XeqY(Req29→release_duration,Rel2→duration), XeqY(Req29→release_start,Rel2→start), XeqC(Req29→assigned_release,2)), XneqC(Req29→assigned_release,2))

'Eco. efficiency (Calc.)' in Rel2: IfThenElse(And(XgteqY(Req30→start,Rel2→start), XlteqY(Req30→end,Rel2→end)), And(XlteqC(Req30→used_devel,2), XlteqC(Req30→used_testers,1), XlteqC(Req30→used_budget,5000), XeqY(Req30→release_duration,Rel2→duration), XeqY(Req30→release_start,Rel2→start), XeqC(Req30→assigned_release,2)), XneqC(Req30→assigned_release,2))

'Eco. efficiency (GUI-input)' in Rel2: IfThenElse(And(XgteqY(Req31→start,Rel2→start), XlteqY(Req31→end,Rel2→end)), And(XlteqC(Req31→used_devel,2), XlteqC(Req31→used_testers,1), XlteqC(Req31→used_budget,5000), XeqY(Req31→release_duration,Rel2→duration), XeqY(Req31→release_start,Rel2→start), XeqC(Req31→assigned_release,2)), XneqC(Req31→assigned_release,2))

'Eco. efficiency (Proposal)' in Rel2: IfThenElse(And(XgteqY(Req32→start,Rel2→start), XlteqY(Req32→end,Rel2→end)), And(XlteqC(Req32→used_devel,2), XlteqC(Req32→used_testers,1), XlteqC(Req32→used_budget,5000), XeqY(Req32→release_duration,Rel2→duration), XeqY(Req32→release_start,Rel2→start), XeqC(Req32→assigned_release,2)), XneqC(Req32→assigned_release,2))

'Modelling an Iconset' in Rel2: IfThenElse(And(XgteqY(Req33→start,Rel2→start), XlteqY(Req33→end,Rel2→end)), And(XlteqC(Req33→used_devel,2), XlteqC(Req33→used_testers,1), XlteqC(Req33→used_budget,5000), XeqY(Req33→release_duration,Rel2→duration), XeqY(Req33→release_start,Rel2→start), XeqC(Req33→assigned_release,2)), XneqC(Req33→assigned_release,2))

'Mysql Backend' in Rel2: IfThenElse(And(XgteqY(Req34→start,Rel2→start), XlteqY(Req34→end,Rel2→end)), And(XlteqC(Req34→used_devel,2), XlteqC(Req34→used_testers,1), XlteqC(Req34→used_budget,5000), XeqY(Req34→release_duration,Rel2→duration), XeqY(Req34→release_start,Rel2→start), XeqC(Req34→assigned_release,2)), XneqC(Req34→assigned_release,2))

'Sqlite Backend' in Rel2: IfThenElse(And(XgteqY(Req35→start,Rel2→start), XlteqY(Req35→end,Rel2→end)), And(XlteqC(Req35→used_devel,2), XlteqC(Req35→used_testers,1), XlteqC(Req35→used_budget,5000), XeqY(Req35→release_duration,Rel2→duration), XeqY(Req35→release_start,Rel2→start), XeqC(Req35→assigned_release,2)), XneqC(Req35→assigned_release,2))

'Costs for refilling (GUI-input)' in Rel2: IfThenElse(And(XgteqY(Req36→start,Rel2→start), XlteqY(Req36→end,Rel2→end)), And(XlteqC(Req36→used_devel,2), XlteqC(Req36→used_testers,1), XlteqC(Req36→used_budget,5000), XeqY(Req36→release_duration,Rel2→duration), XeqY(Req36→release_start,Rel2→start), XeqC(Req36→assigned_release,2)), XneqC(Req36→assigned_release,2))

'Apache Authentication' in Rel2: IfThenElse(And(XgteqY(Req37→start,Rel2→start), XlteqY(Req37→end,Rel2→end)), And(XlteqC(Req37→used_devel,2), XlteqC(Req37→used_testers,1), XlteqC(Req37→used_budget,5000), XeqY(Req37→release_duration,Rel2→duration), XeqY(Req37→release_start,Rel2→start), XeqC(Req37→assigned_release,2)), XneqC(Req37→assigned_release,2))

'DB Auth.' in Rel2: IfThenElse(And(XgteqY(Req38→start,Rel2→start), XlteqY(Req38→end,Rel2→end)), And(XlteqC(Req38→used_devel,2), XlteqC(Req38→used_testers,1), XlteqC(Req38→used_budget,5000), XeqY(Req38→release_duration,Rel2→duration), XeqY(Req38→release_start,Rel2→start), XeqC(Req38→assigned_release,2)), XneqC(Req38→assigned_release,2))

'Auth. Tables' in Rel2: IfThenElse(And(XgteqY(Req39→start,Rel2→start), XlteqY(Req39→end,Rel2→end)), And(XlteqC(Req39→used_devel,2), XlteqC(Req39→used_testers,1), XlteqC(Req39→used_budget,5000), XeqY(Req39→release_duration,Rel2→duration), XeqY(Req39→release_start,Rel2→start), XeqC(Req39→assigned_release,2)), XneqC(Req39→assigned_release,2))

'Auth. DB Communication' in Rel2: IfThenElse(And(XgteqY(Req40→start,Rel2→start), XlteqY(Req40→end,Rel2→end)), And(XlteqC(Req40→used_devel,2), XlteqC(Req40→used_testers,1), XlteqC(Req40→used_budget,5000), XeqY(Req40→release_duration,Rel2→duration), XeqY(Req40→release_start,Rel2→start), XeqC(Req40→assigned_release,2)), XneqC(Req40→assigned_release,2))

'Auth. Pwd Encryption' in Rel2: IfThenElse(And(XgteqY(Req41→start,Rel2→start), XlteqY(Req41→end,Rel2→end)),

And(XlteqC(Req41→used_devel,2), XlteqC(Req41→used_testers,1), XlteqC(Req41→used_budget,5000), XeqY(Req41→release_duration,Rel2→duration), XeqY(Req41→release_start,Rel2→start), XeqC(Req41→assigned_release,2)), XneqC(Req41→assigned_release,2))
'Air caps calculation (GUI-Results)' in Rel2: IfThenElse(And(XgteqY(Req42→start,Rel2→start), XlteqY(Req42→end,Rel2→end)), And(XlteqC(Req42→used_devel,2), XlteqC(Req42→used_testers,1), XlteqC(Req42→used_budget,5000), XeqY(Req42→release_duration,Rel2→duration), XeqY(Req42→release_start,Rel2→start), XeqC(Req42→assigned_release,2)), XneqC(Req42→assigned_release,2))
'Pipe costs (GUI-results)' in Rel2: IfThenElse(And(XgteqY(Req43→start,Rel2→start), XlteqY(Req43→end,Rel2→end)), And(XlteqC(Req43→used_devel,2), XlteqC(Req43→used_testers,1), XlteqC(Req43→used_budget,5000), XeqY(Req43→release_duration,Rel2→duration), XeqY(Req43→release_start,Rel2→start), XeqC(Req43→assigned_release,2)), XneqC(Req43→assigned_release,2))
'PGR & ADG (GUI-Result)' in Rel2: IfThenElse(And(XgteqY(Req44→start,Rel2→start), XlteqY(Req44→end,Rel2→end)), And(XlteqC(Req44→used_devel,2), XlteqC(Req44→used_testers,1), XlteqC(Req44→used_budget,5000), XeqY(Req44→release_duration,Rel2→duration), XeqY(Req44→release_start,Rel2→start), XeqC(Req44→assigned_release,2)), XneqC(Req44→assigned_release,2))
'Special Equipment (GUI-results)' in Rel2: IfThenElse(And(XgteqY(Req45→start,Rel2→start), XlteqY(Req45→end,Rel2→end)), And(XlteqC(Req45→used_devel,2), XlteqC(Req45→used_testers,1), XlteqC(Req45→used_budget,5000), XeqY(Req45→release_duration,Rel2→duration), XeqY(Req45→release_start,Rel2→start), XeqC(Req45→assigned_release,2)), XneqC(Req45→assigned_release,2))
'Special Equipment (Calculation)' in Rel2: IfThenElse(And(XgteqY(Req46→start,Rel2→start), XlteqY(Req46→end,Rel2→end)), And(XlteqC(Req46→used_devel,2), XlteqC(Req46→used_testers,1), XlteqC(Req46→used_budget,5000), XeqY(Req46→release_duration,Rel2→duration), XeqY(Req46→release_start,Rel2→start), XeqC(Req46→assigned_release,2)), XneqC(Req46→assigned_release,2))
'Eco. efficiency (Graphics)' in Rel2: IfThenElse(And(XgteqY(Req47→start,Rel2→start), XlteqY(Req47→end,Rel2→end)), And(XlteqC(Req47→used_devel,2), XlteqC(Req47→used_testers,1), XlteqC(Req47→used_budget,5000), XeqY(Req47→release_duration,Rel2→duration), XeqY(Req47→release_start,Rel2→start), XeqC(Req47→assigned_release,2)), XneqC(Req47→assigned_release,2))
'Eco. efficiency (Proposal GUI)' in Rel2: IfThenElse(And(XgteqY(Req48→start,Rel2→start), XlteqY(Req48→end,Rel2→end)), And(XlteqC(Req48→used_devel,2), XlteqC(Req48→used_testers,1), XlteqC(Req48→used_budget,5000), XeqY(Req48→release_duration,Rel2→duration), XeqY(Req48→release_start,Rel2→start), XeqC(Req48→assigned_release,2)), XneqC(Req48→assigned_release,2))

Table B.10: Example Application: Constraints depending on the release to which a requirement implementation is scheduled.

Part-refinement constraints.

End of 'Collectors' surface'	Max(Req2→end, {Req4→end, Req7→end})
	Max(Req2→assigned_release, {Req4→assigned_release, Req7→assigned_release})
End of 'GUI Preconditions'	Max(Req3→end, {Req0→end, Req1→end, Req33→end})
	Max(Req3→assigned_release, {Req0→assigned_release, Req1→assigned_release, Req33→assigned_release})
End of 'Collectors and installation'	Max(Req6→end, {Req8→end, Req23→end})
	Max(Req6→assigned_release, {Req8→assigned_release, Req23→assigned_release})
End of 'Air caps calculation'	Max(Req9→end, {Req10→end, Req25→end, Req42→end})
	Max(Req9→assigned_release, {Req10→assigned_release, Req25→assigned_release, Req42→assigned_release})
End of 'Pipe Costs'	Max(Req11→end, {Req12→end, Req24→end, Req43→end})
	Max(Req11→assigned_release, {Req12→assigned_release, Req24→assigned_release, Req43→assigned_release})
End of 'PGR & ADG'	Max(Req13→end, {Req14→end, Req26→end, Req29→end, Req44→end})
	Max(Req13→assigned_release, {Req14→assigned_release, Req26→assigned_release, Req29→assigned_release, Req44→assigned_release})
End of 'Costs for refilling'	Max(Req15→end, {Req16→end, Req27→end})
	Max(Req15→assigned_release, {Req16→assigned_release, Req27→assigned_release})
End of 'Special Equipment'	Max(Req17→end, {Req18→end, Req28→end, Req45→end, Req46→end})
	Max(Req17→assigned_release, {Req18→assigned_release, Req28→assigned_release, Req45→assigned_release, Req46→assigned_release})
End of 'Misc. Material'	Max(Req19→end, {Req20→end, Req21→end})
	Max(Req19→assigned_release, {Req20→assigned_release, Req21→assigned_release})
End of 'Economic efficiency'	Max(Req22→end, {Req30→end, Req48→end})
	Max(Req22→assigned_release, {Req30→assigned_release, Req48→assigned_release})
End of 'DB Auth.'	Max(Req38→end, {Req39→end, Req40→end})
	Max(Req38→assigned_release, {Req39→assigned_release, Req40→assigned_release})

Table B.11: Example Application: *Part-refinement* constraints.

Bundling sub-requirements

'Collectors' surface'	
Bundle start	Min(bundle_start,{Req4→start, Req7→start})
Bundle end	Max(bundle_end,{Req4→end, Req7→end})
Latest bundle end	XplusCeqZ(bundle_start,5,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req4→assigned_release,Req7→assigned_release),
'GUI Preconditions'	
Bundle start	Min(bundle_start,{Req0→start, Req1→start, Req33→start})
Bundle end	Max(bundle_end,{Req0→end, Req1→end, Req33→end})
Latest bundle end	XplusCeqZ(bundle_start,16,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req0→assigned_release,Req1→assigned_release), XeqY(Req1→assigned_release,Req33→assigned_release),
'Collectors and installation'	
Bundle start	Min(bundle_start,{Req8→start, Req23→start})
Bundle end	Max(bundle_end,{Req8→end, Req23→end})
Latest bundle end	XplusCeqZ(bundle_start,4,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req8→assigned_release,Req23→assigned_release),
'Air caps calculation'	
Bundle start	Min(bundle_start,{Req10→start, Req25→start, Req42→start})
Bundle end	Max(bundle_end,{Req10→end, Req25→end, Req42→end})
Latest bundle end	XplusCeqZ(bundle_start,6,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req10→assigned_release,Req25→assigned_release), XeqY(Req25→assigned_release,Req42→assigned_release),
'Pipe Costs'	
Bundle start	Min(bundle_start,{Req12→start, Req24→start, Req43→start})
Bundle end	Max(bundle_end,{Req12→end, Req24→end, Req43→end})
Latest bundle end	XplusCeqZ(bundle_start,6,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req12→assigned_release,Req24→assigned_release), XeqY(Req24→assigned_release,Req43→assigned_release),
'PGR & ADG'	
Bundle start	Min(bundle_start,{Req14→start, Req26→start, Req29→start, Req44→start})
Bundle end	Max(bundle_end,{Req14→end, Req26→end, Req29→end, Req44→end})
Latest bundle end	XplusCeqZ(bundle_start,16,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req14→assigned_release,Req26→assigned_release), XeqY(Req26→assigned_release,Req29→assigned_release),
	XeqY(Req29→assigned_release,Req44→assigned_release),
'Costs for refilling'	
Bundle start	Min(bundle_start,{Req16→start, Req27→start})
Bundle end	Max(bundle_end,{Req16→end, Req27→end})
Latest bundle end	XplusCeqZ(bundle_start,5,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req16→assigned_release,Req27→assigned_release),
'Special Equipment'	
Bundle start	Min(bundle_start,{Req18→start, Req28→start, Req45→start, Req46→start})
Bundle end	Max(bundle_end,{Req18→end, Req28→end, Req45→end, Req46→end})
Latest bundle end	XplusCeqZ(bundle_start,9,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req18→assigned_release,Req28→assigned_release), XeqY(Req28→assigned_release,Req45→assigned_release),
	XeqY(Req45→assigned_release,Req46→assigned_release),
'Misc. Material'	
Bundle start	Min(bundle_start,{Req20→start, Req21→start})
Bundle end	Max(bundle_end,{Req20→end, Req21→end})
Latest bundle end	XplusCeqZ(bundle_start,6,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req20→assigned_release,Req21→assigned_release),
'Economic efficiency'	
Bundle start	Min(bundle_start,{Req30→start, Req48→start})
Bundle end	Max(bundle_end,{Req30→end, Req48→end})
Latest bundle end	XplusCeqZ(bundle_start,5,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req30→assigned_release,Req48→assigned_release),
'DB Auth.'	
Bundle start	Min(bundle_start,{Req39→start, Req40→start})
Bundle end	Max(bundle_end,{Req39→end, Req40→end})
Latest bundle end	XplusCeqZ(bundle_start,5,latest_bundle_end)
Bundling	XgteqY(latest_bundle_end, bundle_end)
	XeqY(Req39→assigned_release,Req40→assigned_release),

Table B.12: Example Application: Bundling of sub-requirements.

Generalization constraints

End of 'Authentication'	Min(Req1→end, {Req37→end, Req38→end})
	Min(Req1→assigned_release, {Req37→assigned_release, Req38→assigned_release})
End of 'Database Backend'	Min(Req5→end, {Req34→end, Req35→end})
	Min(Req5→assigned_release, {Req34→assigned_release, Req35→assigned_release})

Table B.13: Example Application: *Generalization* constraints.

Require constraints.

'Collectors' surface (GUI)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req4 →start) XlteqY(Req3→assigned_release, Req4 →start)
'Collectors and installation' requires 'Collectors' surface'	XlteqY(Req2→end, Req6 →start) XlteqY(Req2→assigned_release, Req6 →start)
'Collectors's surface (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req7 →start) XlteqY(Req5→assigned_release, Req7 →start)
'Collectors and installation (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req8 →start) XlteqY(Req5→assigned_release, Req8 →start)
'Collectors and installation (Store)' requires 'Collectors' surface'	XlteqY(Req2→end, Req8 →start) XlteqY(Req2→assigned_release, Req8 →start)
'Air caps calculation' requires 'Collectors and installation'	XlteqY(Req6→end, Req9 →start) XlteqY(Req6→assigned_release, Req9 →start)
'Air caps calculation (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req10 →start) XlteqY(Req5→assigned_release, Req10 →start)
'Air caps calculation (Store)' requires 'Collectors and installation'	XlteqY(Req6→end, Req10 →start) XlteqY(Req6→assigned_release, Req10 →start)
'Pipe Costs' requires 'Collectors and installation'	XlteqY(Req6→end, Req11 →start) XlteqY(Req6→assigned_release, Req11 →start)
'Pipe Costs (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req12 →start) XlteqY(Req5→assigned_release, Req12 →start)
'Pipe Costs (Store)' requires 'Collectors and installation'	XlteqY(Req6→end, Req12 →start) XlteqY(Req6→assigned_release, Req12 →start)
'PGR & ADG' requires 'Collectors and installation'	XlteqY(Req6→end, Req13 →start) XlteqY(Req6→assigned_release, Req13 →start)
'PGR & ADG (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req14 →start) XlteqY(Req5→assigned_release, Req14 →start)
'PGR & ADG (Store)' requires 'Collectors and installation'	XlteqY(Req6→end, Req14 →start) XlteqY(Req6→assigned_release, Req14 →start)
'Costs for refilling' requires 'Collectors and installation'	XlteqY(Req6→end, Req15 →start) XlteqY(Req6→assigned_release, Req15 →start)
'Costs for refilling (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req16 →start) XlteqY(Req5→assigned_release, Req16 →start)
'Costs for refilling (Store)' requires 'Collectors and installation'	XlteqY(Req6→end, Req16 →start) XlteqY(Req6→assigned_release, Req16 →start)
'Special Equipment' requires 'Collectors and installation'	XlteqY(Req6→end, Req17 →start) XlteqY(Req6→assigned_release, Req17 →start)
'Special Equipment (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req18 →start) XlteqY(Req5→assigned_release, Req18 →start)
'Special Equipment (Store)' requires 'Collectors and installation'	XlteqY(Req6→end, Req18 →start) XlteqY(Req6→assigned_release, Req18 →start)
'Misc. Material (Store)' requires 'Database Backend'	XlteqY(Req5→end, Req20 →start) XlteqY(Req5→assigned_release, Req20 →start)
'Misc. Material (GUI)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req21 →start) XlteqY(Req3→assigned_release, Req21 →start)
'Economic efficiency' requires 'Collectors and installation'	XlteqY(Req6→end, Req22 →start) XlteqY(Req6→assigned_release, Req22 →start)
'Economic efficiency' requires 'Air caps calculation'	XlteqY(Req9→end, Req22 →start) XlteqY(Req9→assigned_release, Req22 →start)
'Economic efficiency' requires 'Pipe Costs'	XlteqY(Req11→end, Req22 →start) XlteqY(Req11→assigned_release, Req22 →start)
'Economic efficiency' requires 'PGR & ADG'	XlteqY(Req13→end, Req22 →start) XlteqY(Req13→assigned_release, Req22 →start)
'Economic efficiency' requires 'Special Equipment'	XlteqY(Req17→end, Req22 →start) XlteqY(Req17→assigned_release, Req22 →start)
'Economic efficiency' requires 'Misc. Material'	XlteqY(Req19→end, Req22 →start) XlteqY(Req19→assigned_release, Req22 →start)
'Economic efficiency' requires 'Costs for refilling'	XlteqY(Req15→end, Req22 →start) XlteqY(Req15→assigned_release, Req22 →start)
'Collectors and installation (GUI)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req23 →start) XlteqY(Req3→assigned_release, Req23 →start)
'Collectors and installation (GUI)' requires 'Collectors' surface'	XlteqY(Req2→end, Req23 →start) XlteqY(Req2→assigned_release, Req23 →start)
'Pipe costs (GUI-input)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req24 →start) XlteqY(Req3→assigned_release, Req24 →start)

'Pipe costs (GUI-input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req24 →start) XlteqY(Req6→assigned_release, Req24 →start)
'Air caps calculation (GUI-Input)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req25 →start) XlteqY(Req3→assigned_release, Req25 →start)
'Air caps calculation (GUI-Input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req25 →start) XlteqY(Req6→assigned_release, Req25 →start)
'PGR & ADG (GUI-Input)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req26 →start) XlteqY(Req3→assigned_release, Req26 →start)
'PGR & ADG (GUI-Input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req26 →start) XlteqY(Req6→assigned_release, Req26 →start)
'Costs for refilling (Calculation)' requires 'Collectors and installation'	XlteqY(Req6→end, Req27 →start) XlteqY(Req6→assigned_release, Req27 →start)
'Special Equipment (GUI-input)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req28 →start) XlteqY(Req3→assigned_release, Req28 →start)
'Special Equipment (GUI-input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req28 →start) XlteqY(Req6→assigned_release, Req28 →start)
'PGR & ADG (Calculation) ' requires 'Collectors and installation'	XlteqY(Req6→end, Req29 →start) XlteqY(Req6→assigned_release, Req29 →start)
'Eco. efficiency (Calc.)' requires 'Collectors and installation'	XlteqY(Req6→end, Req30 →start) XlteqY(Req6→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'Air caps calculation'	XlteqY(Req9→end, Req30 →start) XlteqY(Req9→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'Pipe Costs'	XlteqY(Req11→end, Req30 →start) XlteqY(Req11→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'PGR & ADG'	XlteqY(Req13→end, Req30 →start) XlteqY(Req13→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'Special Equipment'	XlteqY(Req17→end, Req30 →start) XlteqY(Req17→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'Misc. Material'	XlteqY(Req19→end, Req30 →start) XlteqY(Req19→assigned_release, Req30 →start)
'Eco. efficiency (Calc.)' requires 'Costs for refilling'	XlteqY(Req15→end, Req30 →start) XlteqY(Req15→assigned_release, Req30 →start)
'Eco. efficiency (GUI-input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req31 →start) XlteqY(Req6→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'Air caps calculation'	XlteqY(Req9→end, Req31 →start) XlteqY(Req9→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'Pipe Costs'	XlteqY(Req11→end, Req31 →start) XlteqY(Req11→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'PGR & ADG'	XlteqY(Req13→end, Req31 →start) XlteqY(Req13→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'Special Equipment'	XlteqY(Req17→end, Req31 →start) XlteqY(Req17→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'Misc. Material'	XlteqY(Req19→end, Req31 →start) XlteqY(Req19→assigned_release, Req31 →start)
'Eco. efficiency (GUI-input)' requires 'Costs for refilling'	XlteqY(Req15→end, Req31 →start) XlteqY(Req15→assigned_release, Req31 →start)
'Eco. efficiency (Proposal)' requires 'Eco. efficiency (GUI-input)'	XlteqY(Req31→end, Req32 →start) XlteqY(Req31→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Collectors and installation'	XlteqY(Req6→end, Req32 →start) XlteqY(Req6→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Air caps calculation'	XlteqY(Req9→end, Req32 →start) XlteqY(Req9→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Pipe Costs'	XlteqY(Req11→end, Req32 →start) XlteqY(Req11→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'PGR & ADG'	XlteqY(Req13→end, Req32 →start) XlteqY(Req13→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Special Equipment'	XlteqY(Req17→end, Req32 →start) XlteqY(Req17→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Misc. Material'	XlteqY(Req19→end, Req32 →start) XlteqY(Req19→assigned_release, Req32 →start)
'Eco. efficiency (Proposal)' requires 'Costs for refilling'	XlteqY(Req15→end, Req32 →start) XlteqY(Req15→assigned_release, Req32 →start)
'Costs for refilling (GUI-input)' requires 'GUI Preconditions'	XlteqY(Req3→end, Req36 →start) XlteqY(Req3→assigned_release, Req36 →start)
'Costs for refilling (GUI-input)' requires 'Collectors and installation'	XlteqY(Req6→end, Req36 →start)

	XlteqY(Req6→assigned_release, Req36 →start)
'DB Auth.' requires	XlteqY(Req5→end, Req38 →start)
'Database Backend'	XlteqY(Req5→assigned_release, Req38 →start)
'Auth. Tables' requires	XlteqY(Req5→end, Req39 →start)
'Database Backend'	XlteqY(Req5→assigned_release, Req39 →start)
'Auth. DB Communication' requires	XlteqY(Req5→end, Req40 →start)
'Database Backend'	XlteqY(Req5→assigned_release, Req40 →start)
'Auth. Pwd Encryption' requires	XlteqY(Req39→end, Req41 →start)
'Auth. Tables'	XlteqY(Req39→assigned_release, Req41 →start)
'Auth. Pwd Encryption' requires	XlteqY(Req40→end, Req41 →start)
'Auth. DB Communication'	XlteqY(Req40→assigned_release, Req41 →start)
'Auth. Pwd Encryption' requires	XlteqY(Req5→end, Req41 →start)
'Database Backend'	XlteqY(Req5→assigned_release, Req41 →start)
'Air caps calculation (GUI-Results)' requires	XlteqY(Req3→end, Req42 →start)
'GUI Preconditions'	XlteqY(Req3→assigned_release, Req42 →start)
'Air caps calculation (GUI-Results)' requires	XlteqY(Req6→end, Req42 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req42 →start)
'Pipe costs (GUI-results)' requires	XlteqY(Req3→end, Req43 →start)
'GUI Preconditions'	XlteqY(Req3→assigned_release, Req43 →start)
'Pipe costs (GUI-results)' requires	XlteqY(Req6→end, Req43 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req43 →start)
'PGR & ADG (GUI-Result)' requires	XlteqY(Req3→end, Req44 →start)
'GUI Preconditions'	XlteqY(Req3→assigned_release, Req44 →start)
'PGR & ADG (GUI-Result)' requires	XlteqY(Req6→end, Req44 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req44 →start)
'Special Equipment (GUI-results)' requires	XlteqY(Req3→end, Req45 →start)
'GUI Preconditions'	XlteqY(Req3→assigned_release, Req45 →start)
'Special Equipment (GUI-results)' requires	XlteqY(Req6→end, Req45 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req45 →start)
'Special Equipment (Calculation)' requires	XlteqY(Req6→end, Req46 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req46 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req30→end, Req47 →start)
'Eco. efficiency (Calc.)'	XlteqY(Req30→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req32→end, Req47 →start)
'Eco. efficiency (Proposal)'	XlteqY(Req32→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req6→end, Req47 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req9→end, Req47 →start)
'Air caps calculation'	XlteqY(Req9→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req11→end, Req47 →start)
'Pipe Costs'	XlteqY(Req11→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req13→end, Req47 →start)
'PGR & ADG'	XlteqY(Req13→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req17→end, Req47 →start)
'Special Equipment'	XlteqY(Req17→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req19→end, Req47 →start)
'Misc. Material'	XlteqY(Req19→assigned_release, Req47 →start)
'Eco. efficiency (Graphics)' requires	XlteqY(Req15→end, Req47 →start)
'Costs for refilling'	XlteqY(Req15→assigned_release, Req47 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req6→end, Req48 →start)
'Collectors and installation'	XlteqY(Req6→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req9→end, Req48 →start)
'Air caps calculation'	XlteqY(Req9→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req11→end, Req48 →start)
'Pipe Costs'	XlteqY(Req11→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req13→end, Req48 →start)
'PGR & ADG'	XlteqY(Req13→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req17→end, Req48 →start)
'Special Equipment'	XlteqY(Req17→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req19→end, Req48 →start)
'Misc. Material'	XlteqY(Req19→assigned_release, Req48 →start)
'Eco. efficiency (Proposal GUI)' requires	XlteqY(Req15→end, Req48 →start)
'Costs for refilling'	XlteqY(Req15→assigned_release, Req48 →start)

Table B.14: Example Application: *Require* constraints.

Incompatible constraints

'Mysql Backend' is incompatible to 'Sqlite Backend'	IfThen(XltC(Req34→assigned_release,3),XeqC(Req35→assigned_release,3)
'Sqlite Backend' is incompatible to 'Mysql Backend'	IfThen(XltC(Req35→assigned_release,3),XeqC(Req34→assigned_release,3)
'Apache Authentication' is incompatible to 'DB Auth.'	IfThen(XltC(Req37→assigned_release,3),XeqC(Req38→assigned_release,3)
'DB Auth.' is incompatible to 'Apache Authentication'	IfThen(XltC(Req38→assigned_release,3),XeqC(Req37→assigned_release,3)
'Apache Authentication' is incompatible to 'Auth. Tables'	IfThen(XltC(Req37→assigned_release,3),XeqC(Req39→assigned_release,3)
'Apache Authentication' is incompatible to 'Auth. DB Communication'	IfThen(XltC(Req37→assigned_release,3),XeqC(Req40→assigned_release,3)
'Apache Authentication' is incompatible to 'Auth. Pwd Encryption'	IfThen(XltC(Req37→assigned_release,3),XeqC(Req41→assigned_release,3)

Table B.15: Example Application: *Incompatible* constraints.