Issues on the Composability of Requirements Specifications for a Product Family

Elena Pérez-Miñana*,
Tim Trew*, Paul Krause •
* *Philips Digital Systems Laboratories, Crossoak Lane, Redhill, Surrey RH1 5HA, United Kingdom*
elena.perez-minana@philips.com, tim.trew@philips .com
• *Department of Computing, University of Surrey, Guildford, GU2 7XH, UK*
p.krause@eim.surrey.ac.uk

Abstract

To develop Requirements Management for Product Families that will satisfy all stakeholders and will promote reuse, we have had to carry out a careful analysis of the various issues related to the composability of the elements of the specifications. Furthermore, we firmly believe that it is essential to design these specifications so that they are amenable to the implementation of a tracking system that links them to all the other artefacts of the product development process (PDP), if platform-based product development and enterprise wide requirements management is to be fully supported. In this paper we present the results of this analysis. These are subject to the specification mechanisms currently used by groups both in Philips Consumer Electronics (PCE) and Philips Semiconductors (PS). Our main conclusion is a clear indication that a full solution is difficult to achieve. We also outline a tentative one whose application is subject to the approval of the project's stakeholders.

1. Introduction

Effective requirements management (RM) is an important issue for the Philips Development organisations. Success in this enterprise is subject to the possibility of being able to track any software artefact up and down the line of development, i.e. from those produced during the requirements phase down to the test cases designed to check the appropriateness of the behaviour shown by the final deliverables. In a first attempt to achieve this goal, we looked at the benefits of transferring the requirements documents to a database of requirements The number of changes that an organisation has to go through in order to make such a move [7] is substantial, therefore a considerable amount of time is necessary in order to accomplish it.

In the first instance, our goal was to device a method(s) that will allow the stakeholders to transfer the requirements as they were found in the documents to a relational database, i.e. the underlying framework of RTM, the preferred RM tool at the time [8].

Paper-based specifications support any amount of diversity, inconsistencies, redundancies, etc. In order to build and populate a database of requirements, it was necessary to take a step back and target our efforts on the improvement of the specification mechanisms used. In the first instance, we had to do a careful revision of the present specifications, list their weaknesses and provide the means to solve them. A thorough study was designed and carried out to identify the problems that had to be tackled.

Section 2 includes a brief description of the method that was followed for the analysis, listing out the issues that need to be solved. Section 3 includes an example of a possible strategy that would enable us to produce "composable" specifications. In Section 4 we present the conclusions we have reached at this stage of the work and the preferred way forward.

2. Issues on the "composability" of Requirements Specifications

A top-down strategy was taken to ensure that all issues that had to be addressed were covered. Keeping in mind our main underlying assumption, i.e. that in the long term, we are aiming towards a solution that will enable the successful specification of Product Families (PF) in the context of a distributed development environment, and in a highly competitive market dominated by a growing product globalisation approach which must be effectively supported.

We have reviewed other work carried out in the area of PF specifications, [3],[4],[9]. We found that, overall, the solutions described by other researchers in the area, involve the integration of more than one specification mechanism. This is one of the reasons we have concentrated our efforts in looking at the best way to integrate the specification mechanisms currently used by the stakeholders and seeing how they could best be adapted for adequate PF specification.

Overall the strategy covered the following steps:

1. Identify the different views that were required to represent the requirements specification. In a multi-site distributed development environment, working groups tend to use different mechanisms to express their requirements, it was therefore necessary to go

through all of them and check the strengths and weaknesses in each case.

2. Consider how product diversity would impact the information presented in each of these views. It is essential to look at this matter as today's way-of-working (WOW) is centred on product family (PF) development.
3. Consider the capabilities that the underlying data representation should have to support this diversity. One of the aims of this work is to enable effective Traceability. In order to do this we must help Philips stakeholders move from paper-based requirements to a database of requirements. The solution offered must support the representation of the diversity that is innate to PF specifications.
4. Consider the degree to which existing techniques satisfy these capabilities. One way to reduce the impact of the changes that will occur is to reuse as much as we can the existing specifications. This will eliminate the costs of training our stakeholders in the use of a new specification mechanism whilst at the same time combating the natural resistance to change that we all experience in our condition as human beings.

2.1.Views of the Specifications

In PCE, it is possible to identify various types of high-level information. These constitute the principal views of the specifications.

1. Functional specifications
   - 1.1 Use cases (U-C): the template used is an adaptation of the definition produced by Alistair Cockburn [2].
   - 1.2 Domain Object Model (DOM): a UML class diagram that contains all the domain objects and the relations that exist between them. This work is carried out in the consumer electronics area, the objects manipulated belong to the TV domain.
   - 1.3 Composite/Elementary functions (CF/EF): a function is considered elementary if the functionality they describe is common within the TV domain and further decomposition does not lead to more required diversity. Functions are considered composite if the functionality they describe is again common within the TV domain, and at least 2 elementary functions are used. Each of them can be linked to a component in the platform architecture underlying the PF.
   - 1.4 Top level state models: these characterize the very highest level of system behaviour, showing which combinations of transitions are always available, given the presence of a particular feature.
   - 1.5 Invariants: these describe chunks of behaviour that remain unchanged regardless of the events the system is processing at the time.
   - 1.6 Atomic requirements.
   - 1.7 Definitions: usually the specification documents include a dictionary section that contains the definition of the domain terms found throughout the document. It provided a valuable initial source of information to build the DOM.
2. User interface (UI) specifications: since the functional specifications are more stable than the UI specifications, the work that has been carried out to this moment has concentrated on the former. At present, it is unclear how much attention should be paid to the representation of the latter

.

These views might be filtered for specific purposes, such as a product-specific view for a regional manager, or represent the full product family diversity, e.g. for an architect. The following section summarises the issues that surfaced during the analysis of the specification mechanisms listed. It concentrated on the following subset of functional specifications: U-C, DOM, and elementary/composite functions (EF/CF) since these required the most sophisticated manipulation to obtain a product-specific view. Many aspects of the other views can be obtained by suitable filtering of a comprehensive model/description.

2.2. Issues that must be resolved to ensure techniques supporting composable specifications

The common characteristic in all the items listed is the that the solution to any of them will only be found if a common plan of action is agreed with all the stakeholders, otherwise it will not be possible to produce a definitive successful approach. For that reason, in certain cases, we have suggested alternative solutions.

1. We consider the U-C as the most effective specification mechanism to express the system requirements at the PF level. The template, which is a tailored version of Cockburn's use case [0], is very flexible and therefore easily adapted to support the specification of diversities. At present, they constitute the representation mechanism that is better understood by the majority of the stakeholders, and it is amenable to database development. However there are a number of slight difficulties that must be tackled before they provide the necessary functionality. Initially, we need to rewrite the expressions that are used to express the flow of events of a U-C. They must be amenable to represent product diversity, and understandable to the users of the specifications. Furthermore, they must be implementation independent due to the high level nature of the PF system requirements.

2. A further issue that must be tackled is in relation to the mapping between the U-C and the other specification mechanisms that describe lower level forms of behaviour, i.e. those of a specific product. One major component of a use case are the pre-/post-condition sections. It is necessary to deal with them at the earliest opportunity. The main issue that needs to be resolved lies in their durability, i.e. the scope of their validity.
3. The stakeholders want to minimise the constraint on the sequencing used in the specifications. This means that, if for a particular U-C the flow of events is mapped, for example, to a sequence of EF invocations ef_1, ef_2, ef_3, it does not mean that at implementation time these EF have to be invoked in that order, because this should be an implementation decision. To handle this requirement, it is necessary to consider the specification of a number of operators that will allow the authors to indicate when EF calls may be carried out in parallel, i.e. when the order of execution is not an issue. There will be situations in which the order has to be adhered to, unless all the EFs are re-written to guarantee that wherever they are invoked the order of execution is not an issue.
4. The UI issue will come up continuously throughout all the specifications, how realistic is it to give feedback to the user information that is not entirely true, although it should be at the end of the execution?
5. How are we going to handle the composition of EFs whose action affect the same variables?, e.g. the TV's Last Status, i.e. the group of TV settings that are changed implicitly by the system.
6. The use cases we have worked on throughout the analysis have highlighted a number of inconsistencies in the specifications with regard to the DOM. This shows the advantage of having a high level view of all the domain objects. It guarantees that all the stakeholders will agree on concepts, naming conventions, relations between domain objects, etc, reducing the mismatches and inconsistencies that are continuously found. The use of a DOM to support the PF development comes from the COPA method that is described in 0.
7. Another problem that surfaced during the review of the composite functions document was the amount of redundancy that exists in them. At least in the present form. The manner in which the CF has been specified is undoubtedly a solution but, there is so much redundancy, it is not the best example of a readable specification. It is important to decide what level of readability the customer is expecting of future specifications.
8. The review process has allowed us to detect a number of chunks of functionality that are clear candidates for invariants. This is something worth considering because this specification mechanism, if it is properly used, could help enormously to reduce the degree of redundancy we have detected in the specifications.
9. It might be useful to include a section that will factor out all the variable names and to which the user may easily refer to, because there are a number of properties, attributes that it is necessary to refer to, even at the highest level of specification possible, e.g. Last Status.
10. It must be possible to express points of variation, either of product diversity of the current product or of inter-feature interaction in a way that the product-specific interactions can be expressed later. These points of variation might exist at any of the three levels, i.e. use case level, elementary function level or composite function level. We need to be clear about the unit of reuse — is it the composite function or the use case? If it is to be the use case then, drawing an analogy with component composition, the points of variability will often have to be exported from lower levels to the use cases "configuration interfaces". However, we may find that there are some cases in which a use case provides enough constraints on the context in which a composite function is to be executed for it to specify the conditions that depict the points of variation.
11. At each level, the points of variation that are exported in "configuration interfaces" must be at a level of granularity that satisfies all lower levels. There are details that are below the concern of the use case and it may be very difficult to specify these conditions in a way that is correct for all lower levels of functional granularity. This provides an argument for minimising the number of levels of granularity.
12. Whereas elementary and composite functions provide reuse of the lowest level of behaviour, top-level state models characterise the very highest. It may be possible to create a general state model, supporting all feature combinations, which can be filtered for the set of features available in a specific model or, if some feature combinations are infeasible, to be able to model individual features and then to compose them within the very highest level state chart. In either case, it may be necessary to add product-specific transitions to the generic framework.

2.3. A strategy towards the generation of "composable specifications"

Our immediate aim is to enable the generation of "composable" specifications in the TV domain. In the previous section, we noted that the lowest level of a product's functionality should be described in terms of EFs and/or CFs. We also mentioned U-Cs as the most appropriate specification mechanism to describe the functional requirements at the PF level. This means that the strategy must include a series of steps that show how to proceed from the top-level U-C based PF specification to the lowest level EF/CF component specification.

The mapping takes as input the flow-of-events section of the U-C and produces as final output the same flow expressed as references to methods and/or objects of the DOM [3] and invocations of EF/CF. In this manner, we are showing how the different specification mechanisms fit together, making sure that product diversity is handled correctly at the different specification levels, that redundancy is minimised throughout, and that the different views that can be obtained through filtering the existing specifications will read as expected. The whole process of creation, maintenance and generation-on-the-fly should be as automatic as possible.

For any U-C contained in the TV-Platform PF specifications [2], the following steps were defined to map its flow-of-events section to the lowest level of functionality, i.e. EF/CF, and references to objects/attributes/methods of the DOM:

1. Extract the flow of events section of the U-C: at this stage no consideration has been given to the pre-, post-condition section, actors or super-ordinate/sub-ordinate use cases. The latter could be used to factor out the functionality thereby providing reusable pieces across the specifications at a latter stage of development. Once the flow of events of the U-C is transformed to a series of EFs calls, each of the calls will be translated into its respective effect section (the main body of an EF) that can be found in a number of requirements specification documents used within PCE.
2. The flow-of-events of the U-C expressed in the imperative style of the EFs effects makes it easier to visualise the type of specification required by the stakeholders. A number of issues surfaced during the mapping process; it is necessary to agree on one solution to each of them:
   2.1. Rewrite the expressions with additional symbols denoting which actions may be done concurrently and which not (an activity diagram type notation).
   2.2. We are setting new values (e.g. new Bass value) it is necessary to make sure the new value is correctly assigned throughout the execution of the use case.
3. The third step in the mapping process involved linking each of the elements mentioned in the effects listed in the expanded flow-of-events of the U-C to objects/methods/attributes in the DOM. Any object referred-to must appear in the DOM. The model is not a static object; it will grow as more features are incorporated to the products of the family under development.

The DOM constitutes the main reference all the artefacts of the PDP should refer to. The types of associations that can be specified, the possibility of expressing constraints through OCL [6] all provide useful instruments to handle the diversity/uniformity that is the trademark of a PF. It has been possible to detect certain inconsistencies between the existing DOM [3] and the specifications as they stand. For that reason, we have updated the DOM as work has progressed. The proposed changes are regularly discussed with the stakeholders; the ones agreed are subsequently reflected in the DOM.

3. Example of a possible strategy towards the generation of "composable specifications"

The functional requirements, i.e. elementary functions, composite functions, presented in this example describe part of the functionality associated with the power control feature of a Generic TV.

These functional requirements may be composed in a variety of ways to provide views on the total set of requirements for either a PF, a representative of a PF, a region, a feature area, or any other meaningful unit of composition in the context of a development project. This example clearly demonstrates how EFs and CFs can be composed to recreate Use Cases, which are one form of presentation of requirements that is relevant to our study.

The contents of **Table 1** depict the flow of events of the use case "switching from Semi-standby or Standby mode to Power ON mode" (use case PM3 in [2]). It is possible to identify two types of diversity in this use case:

- regional diversity, e.g. the steps associated with event 2 are only relevant to the TV sets produced for the NAFTA region.
- PS diversity: certain features of the TV system should behave in a slightly different way depending on whether the final customer is from PS or PCE. This diversity should initially be handled at the PF level and be subsequently propagated to the other specifications.

The use case specified in Table 1 provides all the detail required regardless of the diversity element. For a product manager based in the European region the flow of events would be shorter and easier to understand, if the events that are not relevant to the functionality of a European TV set were not part of the use case. Also the level of detail in each line means that the reviewers need to process a lot of text each time. In this particular case, it was possible to map the flow of events to a number of EF/CF calls that resulted in a simpler and more succinct description as is shown by the contents of Table 2. Each of the EF/CF invoked in the flow of events is specified in Table 3.

Table 1 U-C PM3: Switch from standby or semi-standby mode to power ON mode

Use Case PM3: Switch from Standby or Semi-Standby Mode to Power On Mode.
Status: Common
Region: All

FLOW OF EVENTS
Trigger: The TV Viewer initiates a switch on event.
1. A Warm Data Initialisation shall be triggered.
2. NAFTA Only: If Smart Clock is available, and Smart Clock Mode is set to AUTO:

- The TV shall tune to the channel with Smart Clock set up;
- The Time shall be downloaded;
- If the Smart Clock channel differs from the channel to be finally tuned to, the TV Viewer shall be requested to wait while Smart Clock download is performed.

3. If Child Lock is active and the switch on event has been generated from the Local Keyboard:
4. System Mute shall be set to ON;
5. A 20 second timer shall be started and a message that Child Lock is active shall be displayed on an otherwise blank screen;
6. Diversity 1: As soon as the timer times-out, System Mute shall be set to ON and the TV switched to Standby Mode.
7. The TV shall tune to the last viewed Channel/Programme or external source, unless:

- The switch on event indicated a Channel Number, in which case the TV shall tune to that Channel; or
- The switch on event requested an AV source, in which case the TV shall tune to AV1 media source.

8. If the required channel/programme is locked, as soon as tuning to that channel/programme begins:
9. System Mute shall be set to ON;
10. Video Mute shall be set to ON;
11. Use Case PC1: Enter Access Code shall be called;
12. System Mute shall be set to OFF and Video Mute shall be set to OFF only if access to the blocked item is granted in the previous event.
13. Recall of Last Audio Status shall be triggered;
14. Recall of Last Video Status shall be triggered;
15. Diversity 2: If the Message Setting is ON, the user message shall be displayed; All regions currently, but PS would like the option for this not to happen.
16. The programme/channel number or external source that will be displayed shall be fed back to the Viewer;
17. If a request to wait message was presented to the user, it shall be terminated as soon tuning to the required channel/programme is started.

In each of the function calls listed in Table 2 it is possible to identify two types of parameters:

- Parameters that represent product diversity, e.g. $Region_NAFTA
- Parameters associated to variables that are required to express the product's functionality, e.g. $ChildLockExists

Table 2 flow-o-events of U-C PM3 expressed as EF/CF calls

Trigger: The TV Viewer initiates a switch on event.
INITIALISE_WARM_DATA
SET_SMART_CLOCK($Region_NAFTA)
CHILD_LOCK($ChildLockExists, $PS, $Timer)
SELECT_INIT_CHANNEL
CHANNEL_LOCKED($ChannelLockExists)
RECALL_LAST_AUDIO_STATUS
RECALL_LAST_VIDEO_STATUS
DISPLAY_USER_MESSAGE (! $PS)
The programme/channel number or external source that will be
displayed shall be fed back to the Viewer;

The parameters used (d1, d2 in Table 3) to handle the diversity are functioning as discriminants [5]. Depending on the particular value with which each function is invoked, there will be lines that will not be selected from the database when the specification of a particular product instance is being generated.

The references to objects of the TV model, e.g. Child Lock, Local Keyboard, Audio Last Status, Video Last Status can be mapped to specific path references of the DOM.

The different levels of detail in the specifications are maintained in a database. The example shows that it is possible to define the necessary filters to generate the specifications pitched at the necessary level of detail and targeted to a particular audience.

4. Conclusions

The review/adaptation of the "composable" specification mechanisms is a continuous activity that is carried out through a number of planned workshops, following a rigorous review procedure to ensure there is agreement between the stakeholders.

The diagram in **Figure 1** shows how the information flows from the PF use case specification to the EF/CF calls. It depicts the role the DOM plays in the task of managing the diversity embedded in the specifications. It summarises the process described in Section 2, which shows, it is possible to work out a path that enables the stakeholders to transform the current functional specifications to a form that is amenable to automation, and in which the diversity that is innate to a PF is effectively handled through the DOM and discriminants.

In the process described we have identified and tackled three main issues considered to be the key factors influencing the quality of the specification mechanisms currently used:

- The amount of Natural Language employed to express the effects of the EFs: a greater degree of informality makes it easier to generate inconsistencies.

Table 3 Elementary/Composite functions invoked by U-C PM3

#define CHILD_LOCK(d1,d2,t1) {d1} If Child Lock is active and the switch on event has been generated from the Local Keyboard: {d1} System Mute shall be set to ON; {d1} A t1 second timer shall be started and a message that Child Lock is active shall be displayed on an otherwise blank screen; {d1 && !d2}As soon as the timer times-out, System Mute shall be set to ON and the TV switched to Standby Mode.
#define SET_SMART_CLOCK(d1) {d1} If Smart Clock is available, and Smart Clock Mode is set to AUTO: {d1} - The TV shall tune to the channel with Smart Clock set up; {d1} - The Time shall be downloaded; {d1} - If the Smart Clock channel differs from the channel to be finally tuned to, the TV Viewer shall be requested to wait while Smart Clock download is performed. {d1} If a request to wait message was presented to the user, it shall be terminated as soon tuning to the required channel/programme is started.
#define CHANNEL_LOCKED(d1) {d1} If the required channel/programme is locked, as soon as tuning to that channel/programme begins: {d1} - System Mute shall be set to ON; {d1} - Video Mute shall be set to ON; {d1} - Use Case PC1 shall be called; {d1} - System Mute shall be set to OFF and Video Mute shall be set to OFF only if access to the blocked item is granted in the previous event.
#define DISPLAY_USER_MESSAGE(d1) {d1} If the Message Setting is ON, the user message shall be displayed; #define INITIALISE_WARM_DATA A Warm Data Initialisation shall be triggered.
#define SELECT_INIT_CHANNEL The TV shall tune to the last viewed Channel/Programme or external source, unless: - The switch on event indicated a Channel Number, in which case the TV shall tune to that Channel; or - The switch on event requested an AV source, in which case the TV shall tune to AV1 media source.
#define RECALL_LAST_AUDIO_STATUS Recall of Last Audio Status shall be triggered; #define RECALL_LAST_VIDEO_STATUS Recall of Last Video Status shall be triggered; #define DISPLAY_CHANNEL_NUMBER The programme/channel number or external source that will be displayed shall be fed back to the Viewer;

It is essential to arrive at a level that is easily readable by the users, and at the same time amenable to automatic consistency checks. The agreed level of formality is still under discussion.

- The specification of points of variability: to manage an appropriate level of diversity in the proposal for "composable" specifications the stakeholders must be clear in the following points:
  - The static characteristics of a specific product
  - Interaction with other features
  - Impact on other features
  - Vertical propagation of points of variation from UC to CF to EF: it must be possible to express points of variation, either of product diversity of the current product or of inter-feature interaction in a way that the product-specific interactions can be expressed later.
  - At present we are looking at the work published by Mannion and his group [5]. We hope to apply their ideas on parameterised discriminants to manage the diversity in our specifications.
- Representing temporal constraints: even though the stakeholders would like a specification mechanism as independent as possible from implementation details of any kind, there are situations in which it is necessary to express temporal constraints of some kind. We are currently looking at the following two possible solutions to this problem:
  - through the use of UML activity diagrams
  - through the use of some semi-formal language

This paper describes on-going work. What has been done to this moment has provided us with useful input on the specific problems that need to be solved, in some cases we have identified possible ways of solving them. Nevertheless, given that a major milestone of this process is to arrive at an agreed solution, we are only advancing in very slowly.

10. References

1. America P., Wijgerden J., "Requirements Modelling for Families of Complex Systems", *3rd International Workshop on Software Architecture for Product Families*, March 15-17, Las Palmas de Gran Canaria, Spain, 2000.
2. Cockburn A., "Structuring Use-Cases With Goals", *JOOP*, Sep-Oct 1997.
3. Gomaa H, Reusable Software Requirements and Architectures for Families of Systems, Journal Systems Software, v 28, 1999.
4. Griss ML, "Product Line Architectures", in Component-Based Software Engineering: Putting the Pieces Together, G.T. Heineman ed., Addison-Wesley, May 2001.
5. Krause P, *TV-Platform, Example Functional Requirements Specification for the TV-Platform*, Philips Internal Report 2001.

6. Mannion M, Lewis O, Kaindl H, Montroni G, Wheadon J, "Representing Requirements on Generic Software in an Application Family Model", *ICSR-6, LNCS 1844*, 2000, pp. 153-169.

7. Pérez-Miñana E, Krause P, "Admit it, your requirements deserve to have a life of their own!", *Proceedings of the 6th Philips Software Conference*, Eindhoven, February 2001.

8. Pérez-Miñana E, Krause P, America P, Empowering Requirements for a Product Family, Proceedings of the 5th International Symposium on Requirements Engineering, Toronto, 2001.

9. Weiss Dm, Lai CT, "Software Product Line Engineering: A Family-Based Software Development Process", Addison-Wesley, 1999.

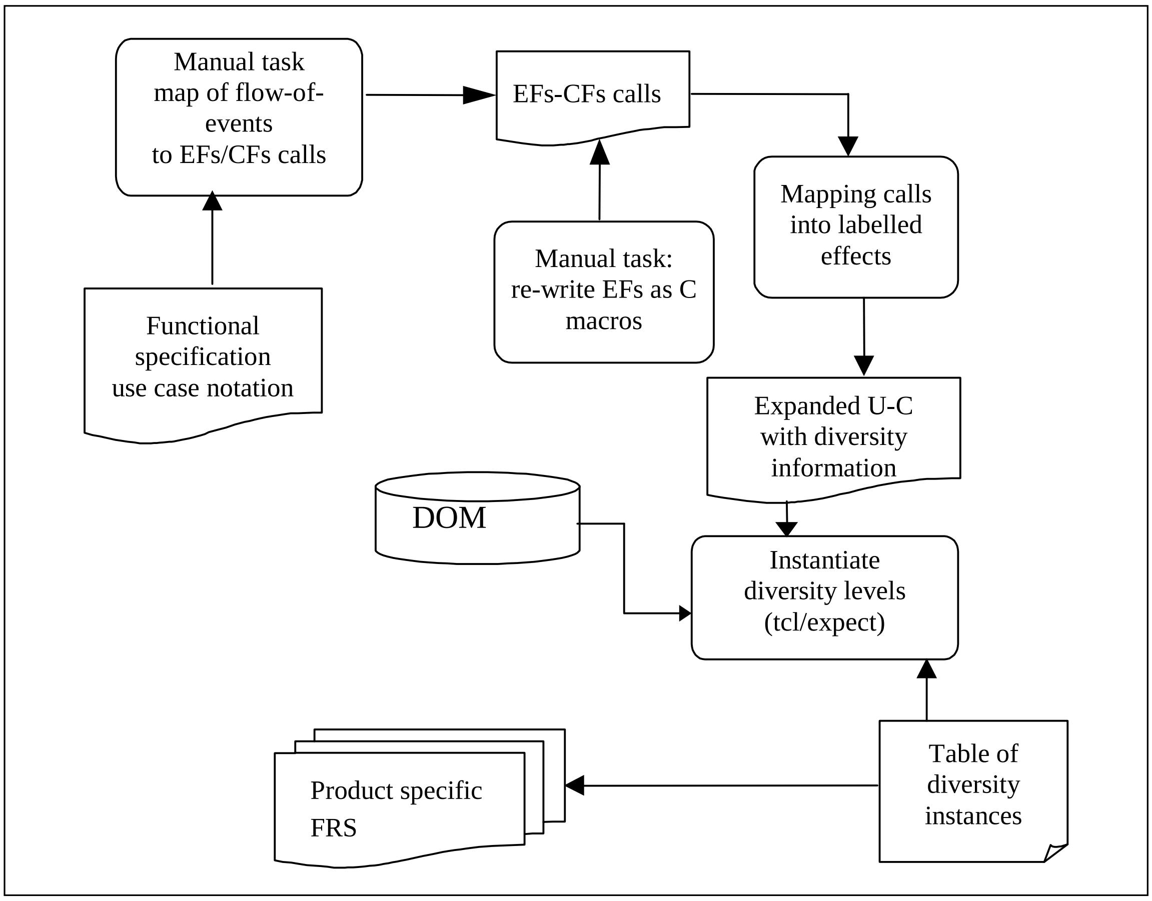
Manual task map of flow-of-events to EFs/CFs calls
EFs-CFs calls
Mapping calls into labelled effects
Manual task: re-write EFs as C macros
Functional specification use case notation
Expanded U-C with diversity information
DOM
Instantiate diversity levels (tcl/expect)
Product specific FRS
Table of diversity instances

Figure 1 DFD of the mapping process from U-C to EF/CF calls