Conflict detection and resolution in distributed design

JOSE´ A. CERONI and ALVARO A. VELÁSQUEZ

Keywords e-Work, collaboration, autonomous agents, protocols

Abstract. As an increasingly common feature, design of today’s complex products and services utilizes distributed collaborative processes. A key factor in the effectiveness of the design processes is conflict management, since conflicts result mainly from designer interaction. Effective support of the design process needs new approaches, from input of specifications to generation of the design documents. These new approaches need to support functions to identify, avoid, and solve conflicts throughout the design process. The research presented in this paper describes a particular approach to conflict detection and resolution in collaborative design processes. The solution proposed lies within the guidelines of e-Work established at the PRISM research group at Purdue University, in that of providing both autonomous agents and active protocols for the distributed design process. The developed Conflict Detection and Management System (CDMS) allows testing for conflict detection and resolution by providing a collaboration interface for the distributed designers. CDMS implements straightforward methods on an Intranet of personal computers platform for detecting and solving the conflicts that otherwise would result in a longer design process. The system also implements a simple learning mechanism for selecting components to include in the design. Experiments performed with CDMS show that product complexity and number of participating designers have a statistically significant effect on the ratio of conflicts solved to conflicts detected, however, only product complexity has a statistically significant effect on design duration.

1. Introduction

Classical design methods for products and services (figure 1) usually consist of the iterative serial execution of tasks, where each task takes care of specific aspects of the design. This approach often resulted in long product and service developing times as well as designs of low quality with inconsistencies, and high cost (Klein 1990). However,
increased complexity, functionality, and modularity present in actual products have turned the design process into an increasingly parallel, concurrent, and collaborative task (figure 2) requiring an effective coordination to solve complex interdependencies among the distributed designers and their particular design tasks (Eberts and Nof 1993, Bonnardel and Sumner 1996, Phillips and Eberts 1998).

Resulting from the collaborative process, conflicts among designer specifications may occur, hindering the design process preventing the distributed design from achieving shorter development cycles or sacrificing quality of the design. Therefore, automated conflict detection and resolution play a key role ensuring effective design processes (Nof 2000, Lara and Nof, 2002). This article presents the development of a computer-based prototype tool for assisting the tasks of collaborative distributed designers (Diaz and Velasquez, 2001).

2. The Conflict Detection and Management System (CDMS)

The main characteristics of the proposed system are:

1. Operation on a local area network.
2. Interface to input specifications by designers.
3. Detection of conflicts based on design specifications.
4. Resolution of conflicts detected (if resolution is not possible, assistance is provided).
5. Learning capability based on designers actions.
6. Adaptability to operational environment (i.e., technology or markets trends).
7. Operation under a supervisor.

These characteristics provide the system with a test bed for experimenting on distributed design. LAN support provides interconnection for designers to enter their design decisions using the system interface. The input operation as described eliminates personal interaction among designers, considered a potential source for conflicts (Klein 1993). The design specification follows a formalism based on the input requirements for the conflict detection and resolution process, as well as characteristics of the design process. Design specification input is in terms of design tasks that allow designers to interact with the system based on entities such as resources, components, or characteristics (Klein 2000). In this way, the description of the product or service under design starts with abstract modules which represent characteristics of the product along with values set as objective for the module attribute (for example, the speed of a car or capacity of a room). Further descriptions provide more detail by limiting the value of attributes, establishing links among modules, decomposing modules into submodules, and specializing modules by defining their classes (figure 3). As figure 3 shows, parent design modules have descendant modules, and a module with no ascendants is the main module (Diaz and Velasquez 2001). Thus, the design structure consists of layers and modular components, allowing specification of objects by unique modules. Designers work collaboratively under the least commitment rule ensuring achievement of global instead of personal objectives.

The system knowledge base comprises three matrices with information on description of components, their crossed compatibility, and learning contribution. Based on the matrices the system relates with the environment by connecting designers preferences, their experience and subjective knowledge with quantitative data of components and their characteristics. Description for each of the information elements is as follows.
2.1. Component description

The description matrix (figure 4) contains the quantitative description of the components. The descriptions allow objective comparison of components alternatives (Diaz and Velasquez 2001). The description matrix is also useful to verify accomplishment of quantitative design objectives.

2.2. Cross component compatibility

A symmetrical matrix represents compatibility among characteristics and components of the design (figure 5). An entry equal to one in the matrix indicates the maximum degree of compatibility between the characteristics or components compared, while an entry equal to zero represents incompatible components or characteristics; finally, an entry equal to two represents unrelated components.

2.3. Learning matrix

Values in the learning matrix (figure 6) are the result of using the system learning function, which relates each component to the objectives of the design. Values range from zero to one, with one representing the maximum satisfaction of the objective under consideration.

2.4. Design objectives

Accomplishment of design objectives relate not only to the product or service itself but also to manufacturing processes, servicing, and recycling due to environmental requirements. Therefore, the design objective will vary depending on the considered customer at the time for the design. However, considering the final customer (consumer) some of the objectives that can be established are ‘durability’, ‘low price’, ‘non toxicity’, and so on (Edwards 1997). It will be the designer’s task to adequately translate these terms into more meaningful specifications (in a quantitative way whenever possible) for the design process. Fuzzy logic provides the means for dealing with qualitative objectives covering a range of possibilities between two extremes (Consortium 1997 and NIBS, 2000). Quantitative objectives on the other hand can be clearly stated and obey logical precepts. The Description Matrix represents quantitative objectives while the Learning Matrix contains qualitative objectives.

Clearly, the goal of the CDMS system is not to automatically generate designs based on given specifications but to support the collaborative work by the designers, making their work more efficient in shorter developing times. As Russell and Norbing (1996) point out, technology has not yet reached the human reasoning and creation capability to automate the design task. Assistance to designers are oriented toward achieving robust designs in shorter developing times, cope with diversity and analysis of great number of variables and alternatives in modern designs (Diaz and Velasquez 2001).

Disregarding the number of objectives to be established for the design, the objectives usually organize in a hierarchical manner. It is possible then to relate objectives, creating a tree relationship among them, with the topmost objective indicating the most important objective to be met by the designers and the parent–child relationships among objectives establishing their particular contribution to the design objective.

Until here, the definition of the CDMS system has dealt with the input of the design specification.
Upon this structure, designers will be able to interact, generating the conflicts to detect and solve by the CDMS system.

2.5. Learning capability

Component detailed specification is not enough to ensure an adequate selection in the final design (Jang et al. 1996). Therefore, the CDMS system requires learning from experience to construct a comprehensive and ever growing knowledge database. Artificial intelligence provides several methods to achieve the learning capability. A combination of fuzzy logic and a particular method of learning instead of neural networks is proposed, based mainly on implementation considerations. The learning method considers designer behaviour and reflects in the values assigned to components in the Learning Matrix. The selection then considers the component, its characteristics, and the design objective to satisfy for the product or service under development. Every time a designer considers but disregards a component, its value in the learning matrix changes according to equation (1). In equation (1), \( n \) represents the number of times the decision has been taken previously, \( x' \) and \( x \) are the learning values after and before the decision is taken.

\[
x' = x \ast \left( 2 - \frac{n}{n+1} \right)
\]

Selection of a component by the designer, changes its value in the Learning Matrix according to equation (2).

\[
x' = x \ast \left( 2 - \frac{n}{n+1} \right)
\]

In this way, learning in the CDMS system occurs as presented in figure 7.

To illustrate the learning process in the system, consider for example that a particular designer proposes component A being part of the design. Upon designer knowledge, the component has a 100% probability of acceptance with a learning value of \( x \). The second time a designer selects component A as part of a design, its learning value increases by \( 3/2 \). However, if a designer considers component A twice but eventually discard it as part of the design, its learning value decreases by \( 1/2 \).

2.6. Conflict detection and resolution

The least commitment modular design detects conflicts before designers enter design specifications into the
system. However, not every conflict appears that early in the design process, making it necessary to implement a detection method. Implementation of such a detection method can take many forms, nevertheless, as Klein (2000) establishes, there is a useful trade-off to consider when selecting a conflict detection mechanism (table 1 and figure 8). The computer-based mechanism has been selected as the alternative for the CDMS system given its conflict detection speed.

The conflict handler of the CDMS system (figure 9) searches, identifies, and solves the conflicts in the design process. Clearly, conflicts vary according to the design process underway, and each detected conflict may have more than one solution possible. Conflict identification considers six conflict types in the CDMS system. Each conflict type relates to specific detection and solution mechanisms as presented next.

2.6.1. Constraint conflict

(a) Detection: Failure meeting some parent module constraints. Constraint conflict requires evaluation and verification of all descendents modules.

(b) Solution: To communicate to the designer the conflict he/she is about to commit and avoid the modification to the design.

2.6.2. Syntax conflict

(a) Detection: A specification by a designer is misunderstand (wrongly compiled) by the system. It may involve language or typing mistakes by the designer.

(b) Solution: Once the system detects the mistake it instructs the designer to re-enter the specification,

<table>
<thead>
<tr>
<th>Alternative</th>
<th>Detection speed</th>
<th>Implementation cost</th>
<th>Update cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Physical</td>
<td>Slow</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>Computer based</td>
<td>Fast</td>
<td>High</td>
<td>Low</td>
</tr>
</tbody>
</table>

Table 1. Trade-off in conflict detection mechanisms.

Figure 8. Design process classification.

Figure 9. Conflict handling in CDMS.
avoiding him/her to continue. An effective interface contributes to avoid this type of conflict by providing choice selection, minimizing designer typing.

2.6.3. Omission conflict

(a) Detection: Achievement of some objectives of the design is not possible if a given characteristic is absent.
(b) Solution: The system reports to the design supervisor the list of missing components and characteristics.

2.6.4. Compatibility conflict

(a) Detection: At input or after editing of a new module, a component results which is incompatible with those already defined.
(b) Solution: The designer must select and input a new alternative component, considering the information in the Compatibility Matrix.

2.6.5. Integrity conflict

(a) Detection: The product or service does not satisfy the minimum requirements to conclude the design process. Absent specifications or components prevent the product or service from being useful for the design objectives. The system checks the list of components and characteristics that should be present in the final design.
(b) Solution: Choosing the module contributing the most towards the design objective based on the design objectives hierarchy (figure 11).

When comparing the original and edited modules, the CDMS system performs the following actions:

- Check for compatibility conflict.
- Check for constraint conflict.
- Verify improvement in design objects.
- For qualitative objectives, measure the contribution of actual and new modules per equation (3):

\[
z = \sum_{i=1}^{m} w_i \cdot x_i
\]

where \(w_i\) represent weights assigned to each contribution \(x_i\) to the qualitative objective \(i\).

- After making the decision to keep or discard the proposed changes to the module, the CDMS system learns according to alternatives presented in table 2.

The module conflict will appear every time designers add or change a specification of any modules, as well as when the design supervisor finishes the process.

Figure 10. Module conflict.
3. CDMs System prototype

A prototype of CDMs developed for personal computer platform using Delphi (Object Pascal) allows distributed designers to collaborate over an Intranet of personal computers using Internet browser interfaces. Figures 12–14 show screenshots of the system (screen interfaces appear in Spanish). The prototype system allows designer interaction in a simple form in order to generate conflicts to test the detection and solving mechanisms. The CDMs system allows performing of the following tasks:

- Identification of the design supervisor.
- Identification of collaborative designers.
- Specification of basic characteristics of the product or service to design.
- Definition of the design objectives and constraints.
- Representation of objects.
- Concurrent participation of designers.
- Conflict detection.
- Conflict resolution.
- Learning from experience.

4. Experiment with CDMS

Experimentation with the CDMS system allowed testing of the system response under different conditions and comparison of them by means of a design of experiment (Diaz and Velasquez 2001). Performance of the CDMS system response by ratios $R_1$ and $R_2$ (dependent variables) measures the effectiveness of the system to avoid design delays due to conflicts and system capacity to solve the conflict found along the design process.

$$R_1 = \frac{\text{Design time}}{\text{Time for finding and solving conflicts}}$$

$$R_2 = \frac{\text{Number of conflicts solved}}{\text{Total number of conflicts found}}$$

<table>
<thead>
<tr>
<th>Case</th>
<th>Original module</th>
<th>Edited module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edited module accepted</td>
<td>$x' = x \cdot \left(2 - \frac{n}{n+1}\right)$</td>
<td>$x' = x \cdot \frac{n}{n+1}$</td>
</tr>
<tr>
<td>Edited module discarded</td>
<td>$x' = x \cdot \frac{n}{n+1}$</td>
<td>$x' = x \cdot \left(2 - \frac{n}{n+1}\right)$</td>
</tr>
</tbody>
</table>

Table 2. System learning (Diaz and Velasquez 2001).
The design of the experiment considers two factors (independent variables), product complexity and number of participating designers. Product complexity, measured by the number of modules the product contains, is considered at high level (26 modules) and low level (17 modules) in the experiment. Number of designers measures the number of people making decisions on the design concurrently and it is considered at high level (nine designers), medium level (six designers), and low level (three designers). Hardware availability limits the number of designers participating in the design concurrently. A factorial experiment based
on the levels of factors in the design of the experiment and considering three replications allow to test CDMS system response in terms of $R_1$ and $R_2$. An analysis of variance performed on the observed $R_1$ and $R_2$, shows that $R_1$ depends on both complexity and number of designers ($P$-value 0.000 for both cases). However, $R_2$ depends only on the product complexity ($P$-value 0.000) and there is no relationship between the number of participating designers and the conflicts solved to conflicts found ratio ($P$-value 0.0749).

5. Conclusions

The research reported presents a model and prototype system to assist modular design by detecting and solving conflicts. The proposed CDMS system allows specification of products or services for both human designers and computer support systems. Designers work in a collaborative network of computer workstations and conflicts result from their interaction delaying the design process. Contribution to generate efficient designs of products or services is the result of identifying and solving conflicts by the system during designer work, shortening the duration of the design process. A design supervisor takes over the conflict solution for cases were the CDMS system is unable to find a solution. The design supervisor is responsible for the design and its conclusion, and his/her participation should decrease as the system experience grows. Design specification of components, characteristics, compatibility, and rate of success of components constitute the CDMS system knowledge base for detecting and solving conflicts during the design process. It is expected that the benefits from using this type of system to be twofold, first, multidisciplinary-distributed design helps to cope with increasing levels of complexity in products and services, and second, better and more robust designs are possible in shorter development times. It is considered that further research in this field should comprise the establishment of more detailed and comprehensive conflict description for detection purposes. Future research should also include algorithmic solutions for most design conflicts for increasing the robustness of solving strategies.

References


DEZS, C., and VElassqueZ, A., 2001, Development of system for conflict resolution in distributed design, Thesis for the Bachelor of Science Degree in Industrial Engineering, School of Industrial Engineering, Catholic University of Valparaíso, Chile.


KLEIN, M., 2000, Towards a Systematic Repository of Knowledge About Managing Collaborative Design Conflicts, Center for Coordination Science, Massachusetts Institute of Technology, Boston, USA.


NOF, S. Y., 2000, Tools and Models for e-Work, Research Memorandum 2000-4, School of Industrial Engineering, Purdue University.

