**RC25108 (W1102-021) February 15, 2011**
**Computer Science**

# IBM Research Report

## A Spectrum of Flexibility - Lowering Barriers to Modeling Tool Adoption

**Doug Kimelman**
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

**Ken Hirschman**
Mindjet

**Research Division**
**Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich**

# A Spectrum of Flexibility – Lowering Barriers to Modeling Tool Adoption

Doug Kimelman
IBM Research
dnk@us.ibm.com

Ken Hirschman
Mindjet
ken.hirschman@mindjet.com

## ABSTRACT

The distinction between whiteboards, free-form authoring with office applications, and formal modeling tools is clear; and the attendant barrier to more widespread use of formal modeling tools is well-conceded. Even experienced modelers need somewhere to splash down ideas, in an unconstrained informal unstructured way, and then to gradually refine, structure, and formalize the content, to ultimately achieve the formal model. Without that, the "deer in the headlights" syndrome is common – users need to somehow divine structured formal conformant model content in order to enter anything, and they simply freeze. At that point, all is lost for a formal modeling tool, and the user reverts to office applications or worse.

This paper highlights two challenges: 1) lowering that barrier to formal modeling, by supporting the streamlined capture of ideas and an unimpeded flow of thought into the modeling tool, and 2) the freedom to move *back and forth* dynamically between more-flexible less-structured modes of operation and modes of operation that are more structured, formal, and rigid. Design and modeling is not a "waterfall" kind of process, in which everything can be done informally and then poured into a formal modeling tool, never to return to working less formally. A user needs to be able to move back and forth. In some cases, activity might even *begin* with a ("legacy") formal model, and proceed to a less-formal derived model for purposes of visualization and stakeholder review.

We discuss experience with production tools that constitute initial steps towards flexible modeling, and we report very positive (albeit anecdotal) feedback from users. As well, we describe possible research directions for future work aimed at advancing this emerging area.

## Categories and Subject Descriptors

D.2.2 [Software Engineering: Design Tools and Techniques – *computer-aided software engineering*]

## General Terms

Design, Human Factors.

## 1. INTRODUCTION

As is discussed in the workshop call for papers [8], formal modeling tools, office applications for free-form authoring, and manual tools like whiteboards, each have their own strengths and weaknesses. There is a need for tools that blend all of these modes of operation to achieve the advantages of each. As we discuss in our abstract above, we hope this will lower some of the barriers to more widespread adoption of modeling tool technology.

Further, as has been evident in previous workshops on flexible modeling tools [7] [8], there are a number of interpretations of the notion of modeling tool flexibility. These include accommodation of:

- totally unstructured input (both text entry and freehand drawing, with sketch manipulation) [7/Sangiorgi and Barbosa] [3],
- unstructured input (both textual and graphical) with automated recognition and formalization [2] [7/Hermans et al.],
- basic visual structuring and free-form authoring (with manual or inferred mapping of structure and style to an underlying more-formal model) [6] [5],
- capture and incorporation of audio and video from design deliberations [7/Nagel et al],
- representations of "narrative" concerning a model, as annotations in the form of graphics, audio, video, URLs, and informal information that is possibly vague and incomplete, all layered over a more-formal representation [7/Breitman et al] [1],
- multiple views / languages in model representation (varying levels of abstraction, complementary aspects of a design, increasing conceptual depth) (proposed by many authors) [8],
- extensible metamodels, and on-the-fly creation of new metamodels (proposed by many authors),
- inference from demonstration vs. specification [8/Yu Sun et al],
- and much more.

The authors of this paper have been involved with production deployments of tools that constitute first steps towards flexible modeling capabilities. We have seen the benefits of such capabilities, as well as their deficiencies and room for improvement. In the following sections, we give brief descriptions of our tools, we discuss our observations concerning the appeal and benefit of the flexible modeling capabilities to our end users, we report very positive (albeit anecdotal) user feedback, and we identify possible follow-on enhancements that could address shortcoming we have observed.

We then suggest that what's needed is an environment in which analysts and designers, moment-by-moment, can choose a degree and form of flexibility that suits their immediate task. Users should be able to move back and forth freely across a spectrum of degrees of flexibility, from "infinite" as on whiteboards (possibly augmented by audio and video capture), to "none" as in rigid formal modeling tools. The prime benefit of totally flexible unstructured input is anticipated to be an unobstructed flow of a user's thoughts into the tool, and the intent is that thinking captured in that way can serve as a starting point for a tool-supported progression towards more-formal artifacts. The prime benefit of rigorous formal modeling has been shown to include automated checking, guidance, and assistance, leading to consistency, correctness, and completeness, resulting in increased quality and productivity. In between those two extremes of the spectrum, there is a need for a continuum of kinds and degrees of flexibility. That continuum would permit gradual refinement of artifacts, and progressively more structure and formality.

Finally, we highlight challenges for achieving these ambitious goals and advancing the emerging field of flexible modeling tools.

# 2. Early Flexible Modeling Tool Features in Production

## 2.1 AWB

Architects' Workbench ("AWB") [1] is a tool used by IT architects for solution design. AWB can be thought of as a sophisticated "classical" formal modeling tool *plus* support for the creative process of "architectural thinking". One manifestation of that support is an emphasis on bridging the gulf between tools for working with informal unstructured information (such as whiteboards and office applications) and tools for formal modeling. With a number of features that in fact constitute first steps towards flexible modeling capabilities, AWB facilitates a gradual progression from initial informal information, to an eventual formal representation of an IT solution design. The initial information is often partial, inconsistent, and vague. The eventual formal representation is a holistic end-to-end model that also includes representation of requirements (functional and non-functional), constraints, rationale, and risk analysis. Below, we describe some of the features of AWB that can be regarded as flexible modeling capabilities.

Although intended strictly as a research prototype, AWB found its way into production use, largely on the strength of demand by practitioners – IBM Global Services architects began using AWB on large-scale engagements for major customers. In justifying their choice of a design tool that was neither a finished product nor officially supported nor regarded as strategic, the architects cited capabilities they found highly compelling. In many cases, that list of capabilities included the flexible modeling capabilities we describe below. Of course, at that time, the architects did not refer to "flexible modeling" per se.

### *2.1.1 Markup and Model*

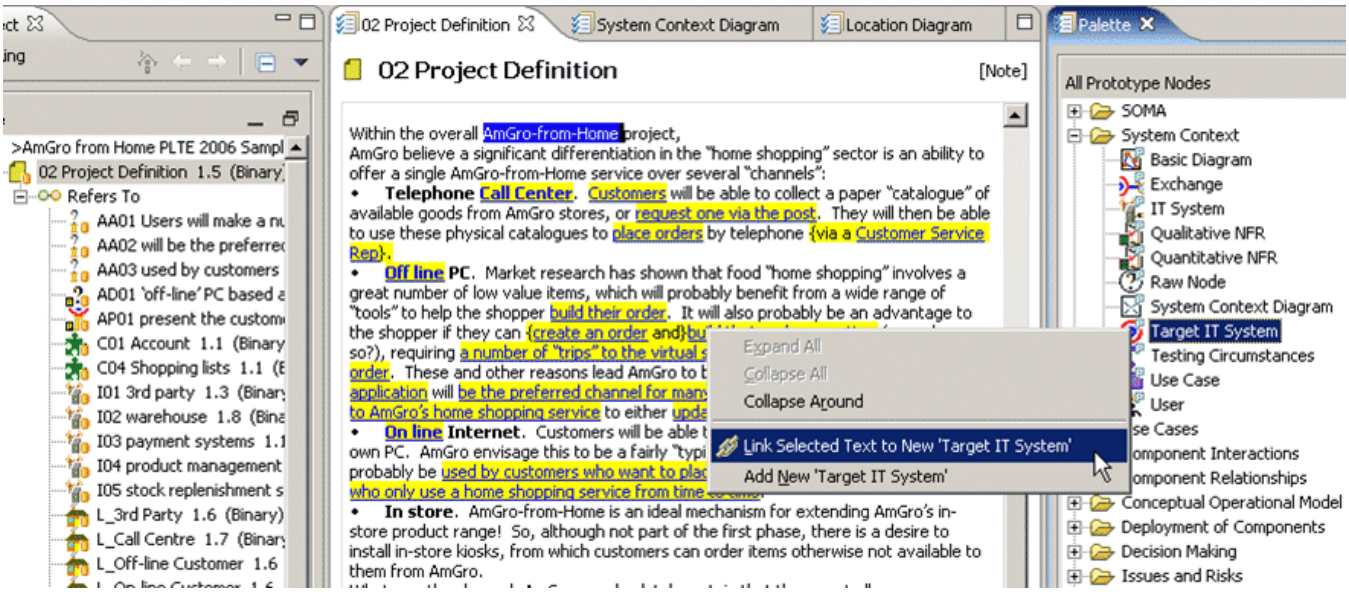
AWB allows entry of free-form text into model elements meant specifically for holding informal textual information, as well as into a 'notes' property that exists in every kind of model element. Users can highlight a passage of text, anywhere they see it in the model, and link it to a newly created model element of a selected type. Throughout the life of the model, users can always traverse the link in either direction: from the informal text to the model element that formalizes it, or from the formal model element back to the text that inspired it. As well, a text passage can be linked to an already-existing model element. If the text exists in the notes property of another existing model element (for example, the notes in a component model element might contain "we should remember to implement the Foo interface as well, in order to satisfy the Bar requirement"), then as well as a hyperlink between the text passage and the target model element, AWB also creates a "semantic link" – a formal relationship between the source and target elements in the formal model. If the metamodel permits more than one type of relationship between the type of the source element and the type of the target element, the user is presented with the possibilities and asked to choose.

This is one example of flexibility in AWB – the ability for users to create both formal and informal content, and to progress from one to the other, maintaining traceability from one form to the other.

Users remarked about two specific uses and benefits of Markup and Model, aside from fundamentally being able to begin informally and progress to more formal:

- It enabled them to capture content live during discussion with stakeholders. They indicated that if they had to do any more demanding structuring, it would have slowed things down and interrupted their train of thought sufficiently to significantly disrupt their discussion with the stakeholder
- It allowed them to rapidly correlate and monitor consistency across the far reaches of very large documents. They found this invaluable in rapidly preparing responses to 1000-page RFPs.

Users often inquired about the possibility of *automated* markup and formalization of new text, based on glossaries built up out of model elements that had already been created. This relates to the proposals of [7/Hermans et al]. And in a more abstract way, this could be seen to relate to the inference of visual style mapping by BITKit [6].




### *2.1.2 Rationale Capture*

The AWB note-taking feature was also used extensively by some users to capture design rationale (e.g. "This server is in the DMZ because we don't trust that the routers imposed on us will handle port forwarding properly; we better be real careful about software firewall setup"). Users might or might not have subsequently formalized such rationale into typed model elements meant to represent architectural decisions, risk analysis, etc. The context for various projects determined whether or not such formalization was seen to be of sufficient value and importance that practitioners were required (or given the time) to be rigorous about it.

In fact, in one case, when a practitioner moved to a new project on which AWB could not be used, he inquired about whether all of the notes in a previous project could be extracted and sent to him, as they constituted critical background and reasoning that he would want to reuse on his new project.

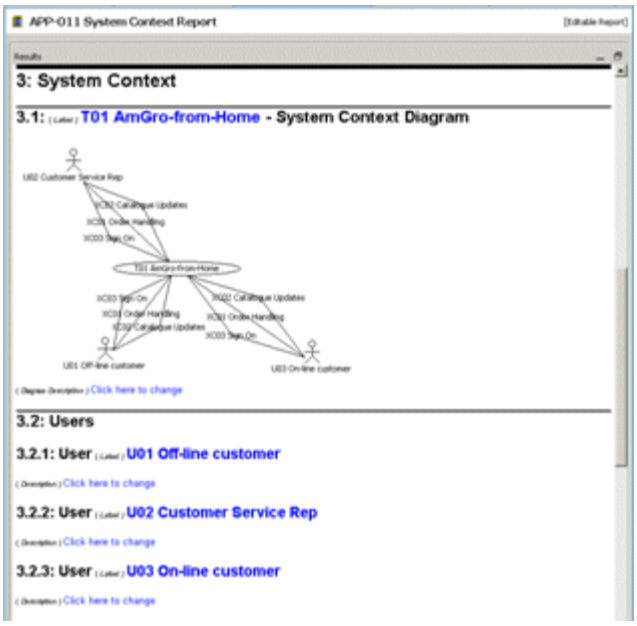
This relates to the importance of flexibility concerning "narrative" as suggested by [7/Breitman et al], and to the support for secondary notation of Chen et al [2].

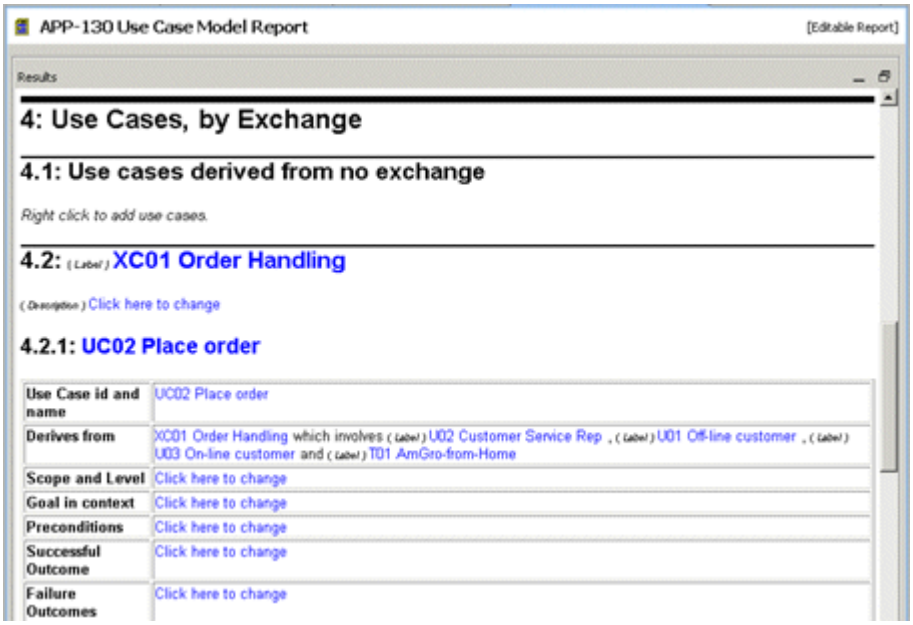
### *2.1.3 Modeling By Forms*

AWB provides automated generation of the textual and graphical design documents that typically are part of the deliverables for IT architecture engagements. The generation is driven by templates defined by a user or provided by a standard library. The templates define content to be extracted from the models, populated into the document, and formatted. Of course, it is often the case that when such documents are provided to a stakeholder, the stakeholder can't resist marking them up and returning them to the architect for revision of the models :-) This gave rise to "Modeling By Forms" in AWB – presenting generated design documents ("forms") as part of the user interface of the tool, and allowing for the tables, lists, and fields in the document to be edited in a manner very similar to that of office applications. Changes made by such editing would be reflected immediately back into the underlying model, and would appear immediately in any other views open on the model at that time. For example, editing a form to add an element to a bulleted list of users of a system, would automatically add a corresponding model element representing that user to the underlying model. Of course, changes made to the underlying model by "classical" means would appear immediately in the form. Forms often contained tables giving properties of model elements, or cells giving lists of relationships to other model elements. Pop-up menus on the form were used to offer model manipulation operations for which there was no obvious corresponding form manipulation.

Allowing users the flexibility to work with their model in a more familiar and convenient fashion turned out to be extremely popular. It made modeling more approachable to analysts who were not expert in sophisticated modeling tools. Furthermore, even for experienced modelers, it allowed certain kinds of work to proceed much more quickly. An unanticipated benefit of the modeling by forms capability was that experienced modelers began defining additional forms as custom views that gathered together parts of the model that allowed them to visually inspect and assure certain kinds of consistency and other global properties of the models they were delivering.

Modeling By Forms became sufficiently important to practitioners, that at one point projects refused to update to a newer release of the prototype in which Modeling By Forms was not yet functional. This early form of flexibility, deployed into production, is similar in some respects to more advanced flexibility affordances being prototyped in BITKit [6]. Although in a different domain, and at a different point on the flexibility spectrum, Chen et al [2] discuss user feedback concerning immediate bi-directional consistency between various renderings of model content.



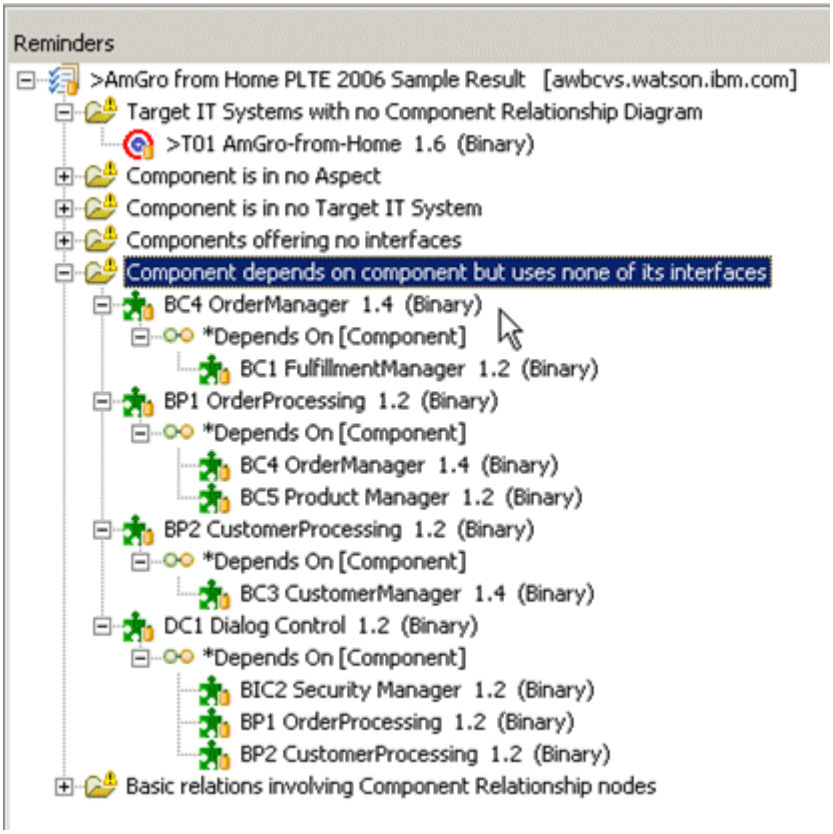

### *2.1.4 Other Forms of Flexibility*

Space limitations preclude detailing other forms of flexibility that are part of AWB, but we will summarize a few briefly.

AWB provides a facility for creating "***Architecture Overview Diagrams***" – arbitrary stakeholder-appropriate juxtapositions of design excerpts taken from across the model, along with informal textual labels and graphic content. These diagrams, laid out manually by a practitioner, do not conform to any diagram/model defined in the metamodel or methodology. They are constructed by the practitioner for purposes of communicating the design to particular stakeholders. Chen at al. [2] point to literature and user feedback suggesting the value of the flexibility to mix elements from different UML diagram types as well as secondary notation. We note that in some sense, this constitutes a progression in "the other direction" along the flexibility spectrum: from more formal and conformant, to more free-form.

In a number of ways, AWB supports early "***roughing in***" of design content, to be followed by progressive refinement via elaboration, structuring, and formalization. AWB permits incomplete non-conformant model fragments to be entered and saved, and it provides ***automated guidance*** (referred to by practitioners as "validation") about possible completions or corrections. An unanticipated outcome of this support was that even experienced architects began to just quickly rough in parts of a design, and then use the automated guidance to

drive their work. We believe that this flexibility and style of work ultimately contributed to greater completeness and consistency of the resulting models.




Another way in which AWB supports early design work is that it allows model elements and relationships to be created initially with very generic types, and subsequently to be refactored to more specific types of model elements from the classical metamodel. We believe that generic types and refactoring permit more convenient exploration and experimentation in the design space, and reduce the impediment to initial capture of modeling thoughts.

Chen et al [2] point to literature and user feedback indicating that "empirical studies show that designers find conventional CASE tools to be overly restrictive, during early design phases in particular" and "diagram editing constraints can be very distracting and off-putting to users, especially during creative design work"; they cite "interference of UML diagramming constraints on creative design in conventional UML case tools"; and they indicate that in their SUMLOW tool they "allow designers freedom to sketch partially-compliant diagrams, with the ability to apply constraint checking during user-requested diagram formalization". SUMLOW is in a different domain and at a different point on the flexibility spectrum, but we believe that these principles apply equally well for AWB as discussed above, and that has been borne out in production use.

### *2.1.5 Conclusions from the AWB Experience*

The significant benefits of the early flexible modeling affordances of AWB have been apparent in practice. Observations and feedback (albeit anecdotal) from on-going production usage confirm the value of those features to users in terms of productivity and quality. Of course, there are other features of AWB, unrelated to flexibility, that also contribute to its effectiveness (e.g. multiple viewpoints, support for a proprietary metamodel, and its general power as a sophisticated formal modeling tool). Nonetheless, the argument can well be made that flexibility is a significant factor in its success.

However, despite on-going success, AWB has not achieved the even more-widespread adoption that might have been possible. Certainly, this is partly due to a lack of support for collaboration. But as well, there were a number of users for whom the barriers to adoption discussed above still remained too high. We believe that a more full spectrum of flexibility, incorporating a number of the forms of flexibility described above, might be able to achieve the next rung on the ladder of widespread deployment.

## 2.2 MindManager

MindManager is a leading commercial tool for "information mapping" [5]. In general, information maps are used to capture, organize and visualize information and ideas, using a hierarchy that is laid out manually in a radial pattern – parent nodes or "topics" are surrounded

by their child nodes or “subtopics”. MindManager facilitates highly streamlined navigation and manipulation of large-scale maps, and provides comprehensive support for collaboration.

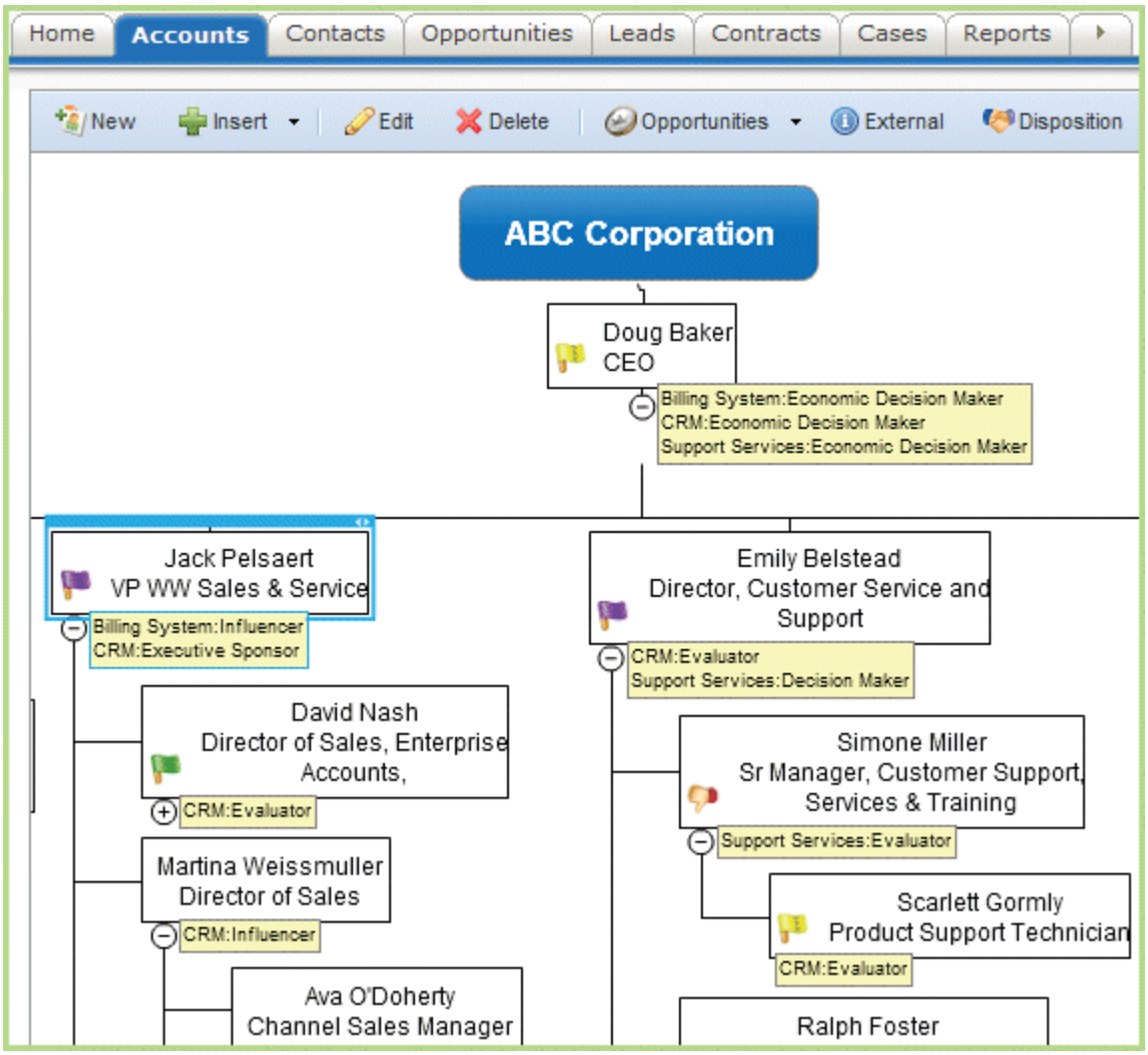
Aside from general information mapping use, and use in specific domains as diverse as project management, business planning, software requirements gathering, and legal document structuring, MindManager has been used as a visual structuring tool integrated with modeling tools to provide capabilities that can be regarded as first steps towards flexible modeling.

Below, we present two applications of MindManager that constitute flexible modeling scenarios.

### *2.2.1 Deal Navigator*

Deal Navigator (“DealNav”) is a MindManager-based add-on to the SalesForce.com (“SFDC”) CRM product [4]. DealNav provides flexibility to an SFDC user by allowing the user to visualize and manipulate the social and organizational structure of a prospective customer using a more convenient and familiar visual structuring paradigm in place of a database query paradigm. DealNav extracts information from the SFDC database and presents it using an alternative layout of hierarchical information – that of an organization chart rather than a radial hierarchy. Model elements can be annotated with text and flagged with various colors to convey information about a person, such as: title, role, influence on purchasing decisions, and sales opportunities. The chart can be manipulated visually or using menus that offer actions relating to organization chart semantics. All changes are reflected back into the SFDC database immediately. Additional links can be established in the chart e.g. to indicate that two customer contacts are acquainted, and those links will be stored in the SFDC database at the end of the session.

The overall capability offered by DealNav is recognized as invaluable by anyone in product sales, or even someone in an industrial research lab scoping out a product division as a new target for technology transfer :-) Note that rather than visualizing and then manipulating an existing database, DealNav could just as easily be used to start fresh with the visual paradigm, entering information about a prospective customer as it is discovered and assembled into an overall structure, and then saved into the SFDC database. Thus, this constitutes an instance of progression in either direction along a spectrum of flexibility, from more flexible to less flexible, or vice versa.



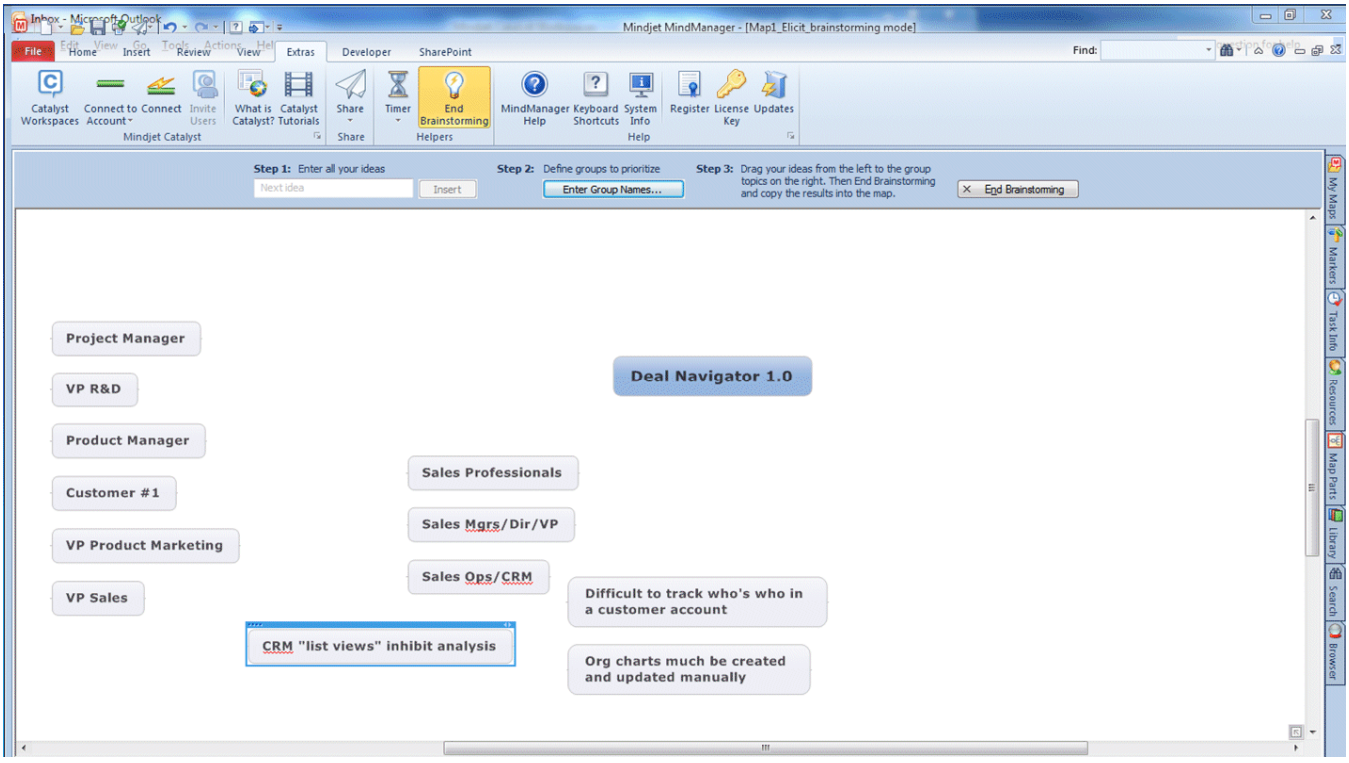
### *2.2.2 Requirements Capture*

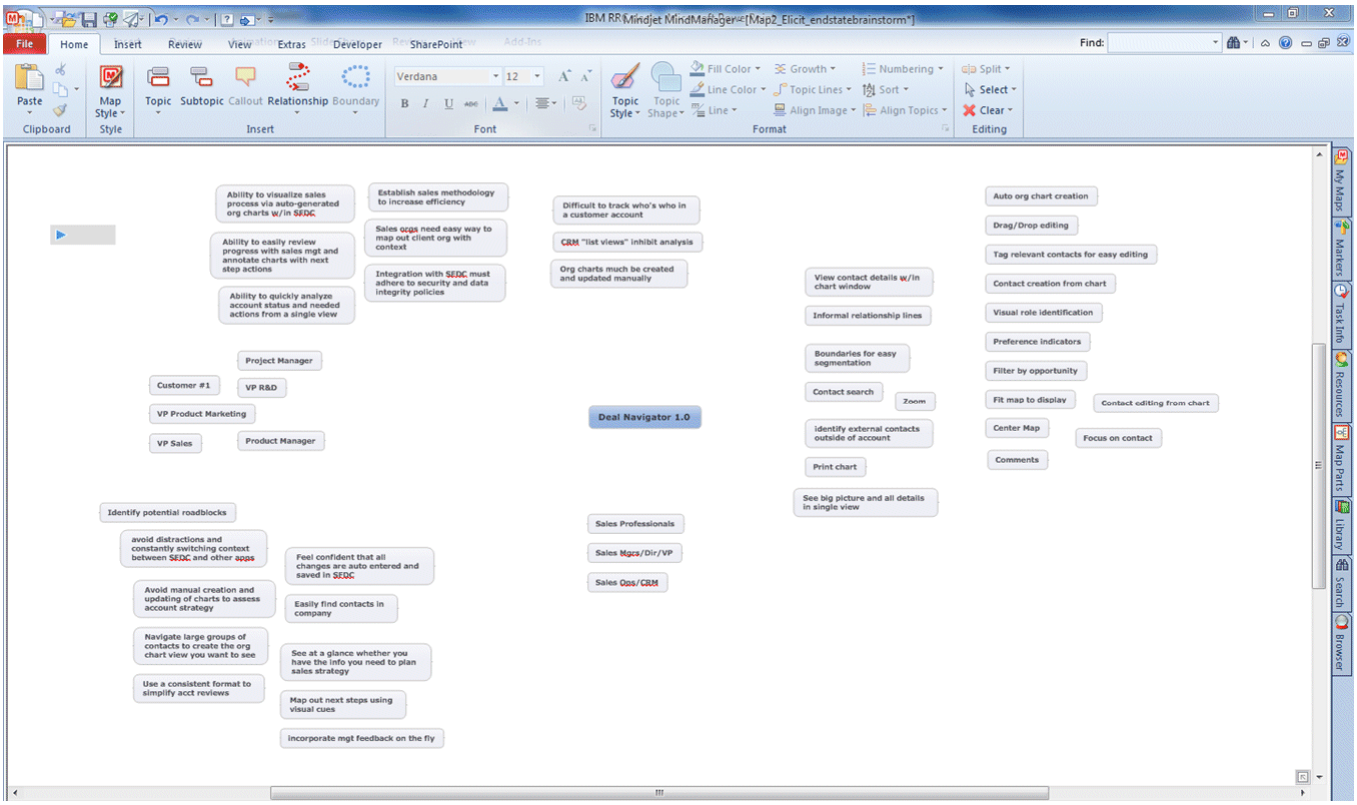
Another flexible modeling scenario based on MindManager involves visual requirements capture and organization, integrated with a downstream requirements modeling tool.

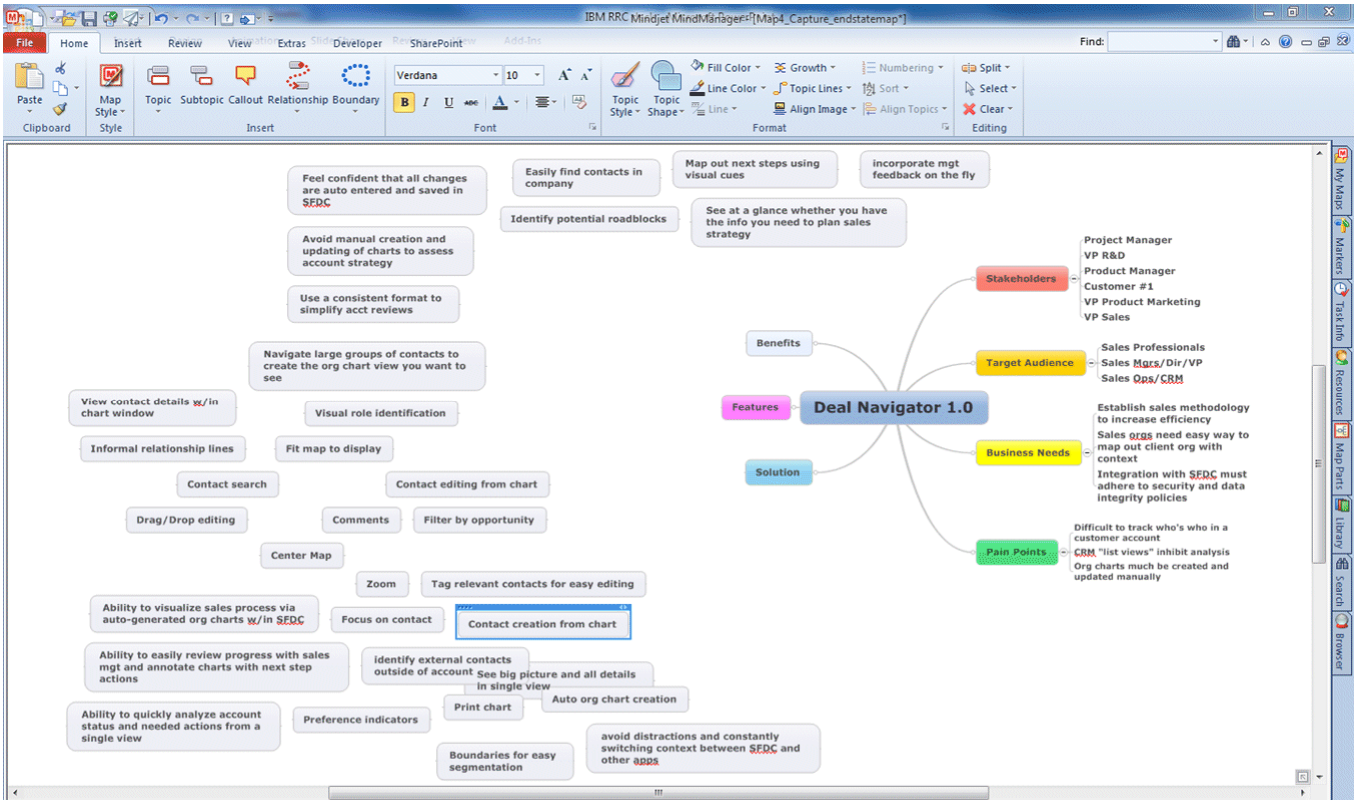
MindManager "brainstorming" mode could first be used collaboratively with stakeholders just to capture thoughts and ideas in as unobtrusive a manner as possible – each time the user enters a text fragment in the "Idea" or "New Topic" field of the user interface and presses the Enter key, the text fragment is simply placed somewhere on the canvas automatically by the tool. When many ideas have been captured, organization of the ideas can begin. A user would set up tags corresponding to topics for grouping ideas e.g. Business Need, Requirement, Feature, Stakeholder, etc., in a tag drawer, and the user would then repeatedly multi-select a number of ideas and click a tag in the tag drawer. When the tag is clicked, all of the selected ideas move into place as child topics of the corresponding group topic. This "mark and move" feature streamlines normal information map operation, to provide an extremely efficient means of organizing project requirements. Elaboration and structuring of requirements would continue with other information mapping capabilities, perhaps involving alternative map layouts that permit ordering, e.g. for prioritization, until eventually all of the requirements have been tagged with tags corresponding to types in the database of the downstream requirements tool, and the requirements can be exported downstream.

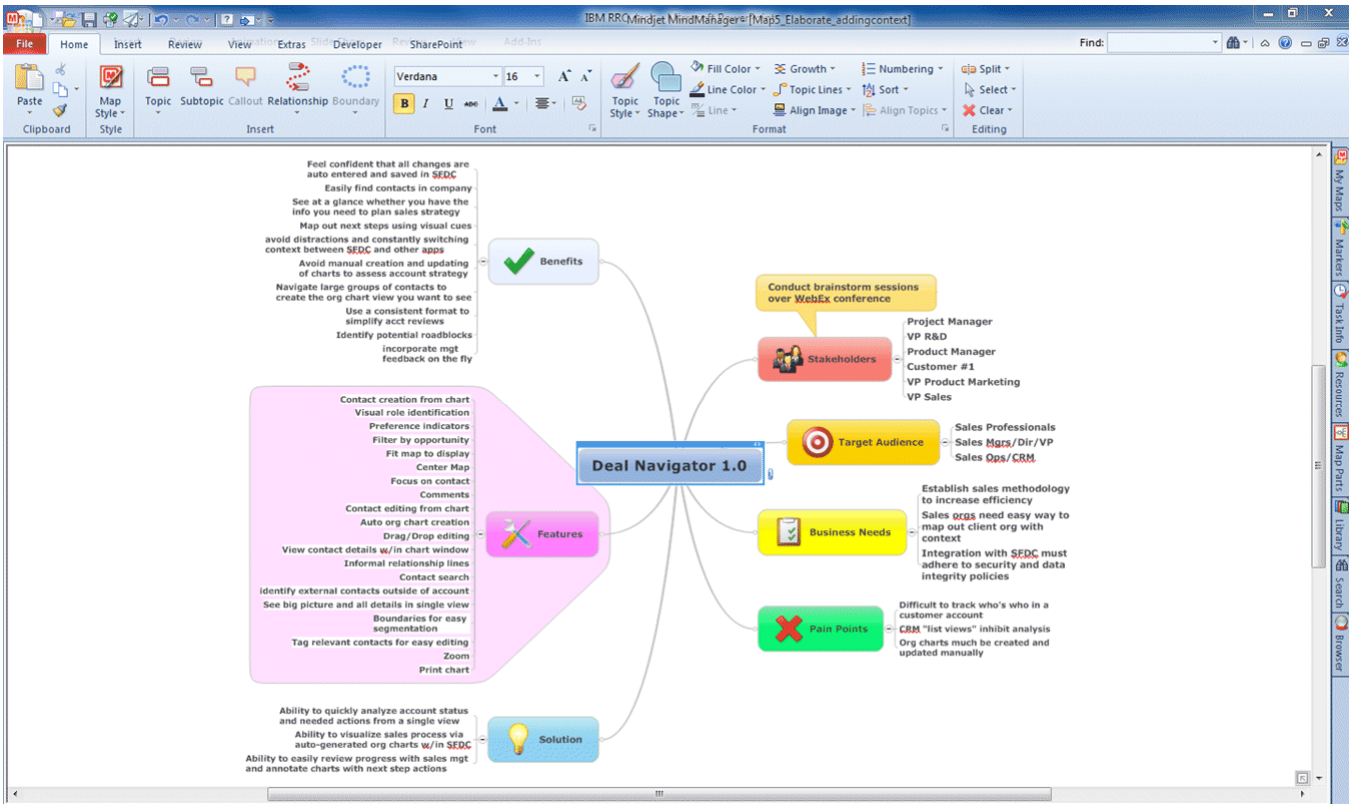
In fact this is the process by which requirements were gathered and defined for the development of the Deal Navigator product.

MindManager can be configured to restrict flexibility so that a user cannot create content that cannot be represented in a downstream database. Below we discuss the challenges of having tools of varying degrees of flexibility and maintaining parallel representations of artifacts.




IBM RRC Mindjet MindManager - [Map2_Elicit_endstatebrainstorm*]
Ability to visualize sales process via auto-generated org charts w/in SFDC
Establish sales methodology to increase efficiency
Sales orgs need easy way to map out client org with context
Integration with SFDC must adhere to security and data integrity policies
Ability to easily review progress with sales mgt and annotate charts with next step actions
Ability to quickly analyze account status and needed actions from a single view
Difficult to track who's who in a customer account
CRM "list views" inhibit analysis
Org charts much be created and updated manually
Project Manager
Customer #1
VP R&D
VP Product Marketing
VP Sales
Product Manager
Deal Navigator 1.0
View contact details w/in chart window
Informal relationship lines
Boundaries for easy segmentation
Contact search
Zoom
identify external contacts outside of account
Print chart
See big picture and all details in single view
Auto org chart creation
Drag/Drop editing
Tag relevant contacts for easy editing
Contact creation from chart
Visual role identification
Preference indicators
Filter by opportunity
Fit map to display
Contact editing from chart
Center Map
Focus on contact
Comments
Identify potential roadblocks
avoid distractions and constantly switching context between SFDC and other apps
Avoid manual creation and updating of charts to assess account strategy
Navigate large groups of contacts to create the org chart view you want to see
Use a consistent format to simplify acct reviews
Feel confident that all changes are auto entered and saved in SFDC
Easily find contacts in company
See at a glance whether you have the info you need to plan sales strategy
Map out next steps using visual cues
incorporate mgt feedback on the fly
Sales Professionals
Sales Mgrs/Dir/VP
Sales Ops/CRM


IBM RRC Mindjet MindManager - [Map4_Capture_endstatemap*]
Feel confident that all changes are auto entered and saved in SFDC
Easily find contacts in company
Map out next steps using visual cues
incorporate mgt feedback on the fly
Identify potential roadblocks
See at a glance whether you have the info you need to plan sales strategy
Avoid manual creation and updating of charts to assess account strategy
Use a consistent format to simplify acct reviews
Navigate large groups of contacts to create the org chart view you want to see
View contact details w/in chart window
Visual role identification
Informal relationship lines
Fit map to display
Contact search
Contact editing from chart
Drag/Drop editing
Comments
Filter by opportunity
Center Map
Zoom
Tag relevant contacts for easy editing
Ability to visualize sales process via auto-generated org charts w/in SFDC
Focus on contact
Contact creation from chart
Ability to easily review progress with sales mgt and annotate charts with next step actions
identify external contacts outside of account
See big picture and all details in single view
Print chart
Auto org chart creation
Ability to quickly analyze account status and needed actions from a single view
Preference indicators
Boundaries for easy segmentation
avoid distractions and constantly switching context between SFDC and other apps
Benefits
Features
Solution
Deal Navigator 1.0
Stakeholders
Project Manager
VP R&D
Product Manager
Customer #1
VP Product Marketing
VP Sales
Target Audience
Sales Professionals
Sales Mgrs/Dir/VP
Sales Ops/CRM
Business Needs
Establish sales methodology to increase efficiency
Sales orgs need easy way to map out client org with context
Integration with SFDC must adhere to security and data integrity policies
Pain Points
Difficult to track who's who in a customer account
CRM "list views" inhibit analysis
Org charts much be created and updated manually




### *2.2.3 Visual Structuring Tools in Flexible Modeling*

In both of the applications above, MindManager is used as a visual structuring tool that is integrated with a downstream modeling tool, in much the same way as is anticipated for BITKit [6]. But where the primary structuring paradigm for BITKit includes lists, tables, and graphs, the primary structuring paradigm for MindManager is radial hierarchies. Furthermore, where BITKit employs automated inference of mapping of visual styles to types of the underlying database schema or metamodel, MindManager employs solely manual tagging.

Visual structuring tools excel at individual or collaborative organization and distillation of information, particularly where there is iterative refinement and there needs to be a concrete deliverable output of the process.

Nonetheless, it seems from experience with AWB, MindManager, and BITKit, that for some users, even basic visual structuring can be too much of an impediment during initial capture of thinking or capture of presentations of large amounts of information. MindManager radial hierarchies, and BITKit shapes, lists, tables, and graphs are not “form-free”. There still is some form or structure. It’s just less complex structure than with full-scale modeling tools. Nonetheless, even just that small amount of structure requires some decision by the user concerning the visual structure or appearance of each fragment prior to it being placed onto the canvas, and it requires some thought by the user to accomplish the necessary keyboard actions, mouse actions, or gestures. That “second order” thought is an impediment – it disrupts the primary train of thought, it is an obstruction that impedes the flow of thoughts and ideas into artifacts, and it can stifle creativity… even for many experts. Chen et al [2] point to studies that have shown that “designers reject tools employing conventional mouse-driven click-and-edit operations during early design-phase work”.

To be even more effective, tools can provide totally free-form input. AWB provides totally free-form text entry. MindManager provides brainstorming mode. Calico provides completely unfettered drawing. SUMLOW is close to this ideal, but it does require that UML model elements be drawn in one of a number of certain ways. Of course, the benefit of that approach is that it can then automate formalization of sketched input, facilitating the progression to the next point on the spectrum. It’s a trade-off.

## 3. Challenges

In our experience with early production deployments, flexible modeling has already been shown to be effective. However, as we discussed above, to achieve the next step in more widespread usage of modeling tools, there is a need for more of a continuum – a broad spectrum of degrees and forms of flexibility, structure, and formality for a user to choose from. As well, there is a need for the user to be able to move back and forth freely between the points on the spectrum, as dictated by the user’s immediate task.

Above, we discussed a number of forms and degrees of flexibility. These could all constitute different points along a flexibility spectrum.

Challenges for achieving these ambitious goals include: maintaining "parallel representations", and "measuring impedance".

## 3.1 Parallel Representations

"Parallel representations" is a vexing long-standing problem for tools that maintain multiple representations of a model, each with its own degree of flexibility, structure, or formality. All such tools face a common set of problems:

- maintaining the correspondence between the representations when changes are made to one of them, including when elements are changed, added or deleted from one representation
- maintaining information in one representation that has no counterpart in other representations.

Anyone working on tools for Model-Driven Development or tools that generate editable source code (Java, Cobol, C) from higher-level or domain-specific languages are well-familiar with such problems.

Chen et al [2] identify some issues they address in SUMLOW e.g. "bi-directional consistency" – changes in a formalized diagram, such as the position of an element, are reflected back into the informal sketched diagram. As well, they identify issues they leave for future work e.g. when an element is added to the formalized diagram, how to synthesize and render a corresponding element in the informal diagram; as well, they leave open the issue of *importing* from downstream modeling tools e.g. for purposes of alternate visualization, interactive exploration with stakeholders, or experimenting with variations.

Deal Navigator does allow addition and deletion of elements in either the organization chart visualization, or the SalesForce.com database, but they disallow addition of information to organization charts that cannot be represented in the database. Any "secondary notations", as they're referred to by Chen et al [2], would be lost because only the database is persisted; the visualization is discarded at the end of each session.

AWB Modeling By Forms and Architecture Overviews do keep parallel representations consistent, but there are restrictions concerning the kinds of changes that are possible in some of the representations.

## 3.2 Measuring Impedance

A different kind of challenge relating to streamlined capture of ideas and an unimpeded flow of thought into the modeling tool is how to measure the "impedance" that a tool offers. Can guidelines be established giving experimental design and frameworks for assessing flexible modeling tools, to measure and compare how streamlined they are for a given set of tasks?

## 3.3 Single Tool vs. Integrated Tools

And, independent of these questions, an interesting question for tool designers to consider is: Can a spectrum of flexibility be achieved by a single tool, perhaps with some explicit user controls over degrees and forms of flexibility? On the other hand, can these goals be achieved effectively by integration of a number of distinct but interoperating tools?

## 4. Conclusion

We eagerly anticipate the advancement of the emerging field of flexible modeling tools, addressing such challenges, and eventually fulfilling the full promise of modeling tool technology, with widespread deployment into production use.

## 5. REFERENCES


[1] Abrams, S., Bloom, B., Keyser, P., Kimelman, D., Nelson, E., Neuberger, W., Roth, T., Simmonds, I., Tang, S., and Vlissides, J. 2006. Architectural thinking and modeling with the Architects' Workbench. *IBM Systems Journal* 45(3), 2006, 481-500.

[2] Chen, Q., Grundy, J.C., and Hosking, J.G. 2008. SUMLOW: Early Design-Stage Sketching of UML Diagrams on an E-whiteboard. *Software – Practice and Experience* 38(9), July 2008, 961-994.

[3] Mangano, N., Baker, A., Dempsey, M., Navarro, E. and van der Hoek, A. 2010. Software Design Sketching with Calico. *Proc. 2010 IEEE/ACM Conf. on Automated Software Engineering*, Sep 2010, 23-32.

[4] Mindjet Deal Navigator Overview. *http://mindjet.com/ products/add-ons/deal-navigator/overview*. Accessed 2011-02-07.

[5] Mindjet MindManager. *http://mindjet.com/mindmanager-uses/brainstorming*. Accessed 2011-02-07.

[6] Ossher, H., Bellamy, R., Simmonds, I., Amid, D., Anaby-Tavor, A., Callery, M., Desmond, M., de Vries, J., Fisher, A.and Krasikov, S. 2010. Flexible modeling tools for pre-requirements analysis: conceptual architecture and research challenges. *Proc 2010 ACM Conf. on Object oriented programming systems languages and applications*, Oct 2010, 848-864.

[7] Program of ICSE 2010 Workshop on Flexible Modeling Tools. *http://www.ics.uci.edu/~tproenca/icse2010/flexitools /program.html.* Accessed 2011-02-07.

[8] Program of SPLASH 2010 Workshop on Flexible Modeling Tools. *http://www.ics.uci.edu/~nlopezgi/flexitools/ program.html.* Accessed 2011-02-07.

[9] Call for Papers of ICSE 2011 Workshop on Flexible Modeling Tools. *http://www.ics.uci.edu/~nlopezgi/* flexitoolsICSE2011/cfp.html. Accessed 2011-02-0