User-Centered Design and Business Process Modeling: Cross Road in Rapid Prototyping Tools

Noi Sukaviriya[1], Vibha Sinha[2],
Thejaswini Ramachandra[3], Senthil Mani[2], and Markus Stolze[1]

[1] IBM TJ Watson Research Center, Hawthorne, NY
[2] IBM India Research lab, Delhi, India
[3] IBM Software Group, Bangalore, India
{noi, mgstolze}@us.ibm.com,
{vibha.sinha, thejaswini, sentmani}@in.ibm.com

Abstract. Fast production of a solution is a necessity in the world of competitive IT consulting business today. In engagements where early user interface design mock-ups are needed to visualize proposed business processes, the need to quickly create UI becomes prominent very early in the process. Our work aims to speed up the UI design process, enabling rapid creation of low-fidelity UI design with traditional user-centered design thinking but different tooling concepts. This paper explains the approach and the rationale behind our model and tools. One key focal point is in leveraging business process models as a starting point of the UI design process. The other focal point is on using a model-driven approach with designer-centered tools to eliminate some design overheads, to help manage a large design space, and to cope with changes in requirements. We used examples from a real business engagement to derive and strengthen this work.

Keywords: User-centered Design Process, UI Design, Model-driven User Interface, Low-fidelity UI Tools.

1 Problem Statements and Context

In the world where competition for business is severe, we are in need for better tooling to support the solution design process. One such design activity is the user-centered design process. Whilst user observations and interviews are essential part of the process, these activities are undoubtedly irreplaceable by IT tools. However, some other parts of the user-centered design process – in particular coming up with low-fidelity UI mock-ups in the early design phase – is still lacking by way of appropriate tools. Many popular tools designers use today such as Adobe Illustrator, Adobe Photoshop, or Microsoft PowerPoint, were developed for other purposes such as drawings and presentations. They support high-fidelity drawings in nature and require high-fidelity efforts to produce low-fidelity mock-ups. This class of tools also lacks support for information structure and design management, hence dealing with requirement changes throughout a project is rather painstaking. Another class of tools such as Macromedia Dreamweaver popular for creating web-based interfaces has more support on data connections and back end services, but the tooling itself is yet another high-fidelity tool. Some research tools were explored such as SILK [1] and DENIM [2] for capturing hand sketches low-fidelity UI and navigation structure. However, the outputs of these tools are not sufficiently organized visually, hence are not practical for business practices. None of these tools mentioned allow a quick way to come up with low-fidelity mock-ups with support for proper alignment and visual organization.

Many people often associate low-fidelity mock-ups with low-cost, hand-drawn interfaces. While paper and pencil are the most used tools, designers use other tools in conjunction [3]. Our preliminary interviews with several user-centered designers revealed that hand-drawn sketches were not used when they presented their UI mock-ups. They always start out with hand sketching but that typically lasted shortly. Some mentioned that their hand-drawn sketches did not look organized enough. But most agreed that it would take an even larger effort to use hand-drawn sketches including re-drawing repeatedly and scanning paper sketches. Needless to say that maintaining hand-drawn sketch changes even in the early design phase for a large project is far from viable. There's a level of visual organization and alignments, implicitly required in presenting UI design ideas in the early design phase, especially in information intensive applications. Without sufficient visual organization, the review process will suffer. All of the designers we interviewed commonly use high-fidelity tools mentioned above to create mock-ups that look low in fidelity.

In IT consulting practices, many business engagements nowadays often start with business process modeling – be it for an entirely new business process or transforming an old process to a new and improved one. Within our company, while Microsoft Visio is still the most popular tool for business process modeling, many consultants now have switched to use an IBM software product called Websphere Business Modeler (WBM.) In this kind of engagements, business analysts use WBM to draw business process diagrams, which are then used as a way to propose and have the customers approve the work. Prompt demand for low-fidelity UI mock-ups is three-fold in such engagements. First is an earlier need even before the UCD process begins. Low-fidelity UI designs and storyboards are commonly used as a means to visualize how a business process might work. Visualization of connected boxes as usually diagrammed in a business model has limitations; they do not give people a good grasp of a more tangible understanding of the process. Secondly, once the work is approved, low-fidelity user interface mock-ups are needed as an instrument for discussing detailed system design and requirements. Throughout the early design phase, business analysts, subject matter experts, architects, and UCD designers gradually and repeatedly discuss over different parts of the UI design to comb out detailed requirements, conflicts of understanding, and realization of the need for further investigations. The UI design repeatedly changes over the course of this activity. The essence of using low-fidelity mock-ups as a way to share the understanding of a solution is monumental in our experience. Lastly, once the design starts to firm up, user-centered designers review low-fidelity UI mock-ups with the users for early design feedback. This has to be done as quickly as possible before turning the design into high-fidelity as the development team is often ready and waiting for inputs at this stage.

The needs for UI mock-ups early in the design phase do not require high fidelity really. That is, not all buttons have to be in the right place, not all UI components have to be in their final locations. The mock-ups do not need finished styles and look. The mock-ups are used for the purpose of understanding the process. For business executives who view the UI with a business process, the low-fidelity mock-ups merely gives them another, yet very powerful, perspective to the process flow and interactions between people in the process. For business analysts and subject matter experts (SMEs), low-fidelity UIs are just a matter of making sure the required business functions are supported and information is presented with the proper semantics. For the end users, designers use low-fidelity UI mock-ups to solicit their feedback. Designers focus on whether the users can understand the information presented, whether they can perceive what needs to be done, whether some information is missing, and whether certain information is not necessary. We do not want to present high-fidelity UI design to these various reviewers as they may be caught in unnecessary cosmetic discussions [4].

We hope to have convinced the readers at this point of the increasing demand for quick low-fidelity UI mock-ups and their business contributions early in the design process. Lack of proper tooling support for quick UI sketches that can change frequently and rapidly would enhance the requirement gathering process. Our focus is on a fast UI mock up tool that leverages business process models that helps maintain the UCD perspectives. We also focus on helping designers manage a large design space. Our other goal not emphasized in this paper is to support the end-to-end UI design process from low fidelity to high fidelity design to UI code generation. This latter endeavor requires a clean separation of concerns from people with different skill sets, i.e. UCD designers and developers, as moving from design to development increasingly adds technical complexity. Our ultimate goal is to encompass within our model and tools sufficient support for low fidelity design thinking and management, as well as the high fidelity UI production process while maintaining the overall usability of the tools.

2 Related Work

Recent related research with a similar goal to our work of providing UCD support is in by Campos and Nunes in [5]. In this work, they built a user-centered tool for rapid prototyping using the notations for abstract prototypes developed by Constantine [6]. Their focus is on using interaction patterns to fill in design blocks while our work fills in design blocks with connections to business data elements, users' read/write access to information as restricted by the business process, and UI related semantics called upon by the designer.

Automatic generation of user interfaces dated back to the early 90's. In [7], a comprehensive notation for expressing the quality of data attributes was used. The automatic generation effort in this work attempted to lay out page detail as well as a page's grouping and semantic structure; the latter task is proved to be quite easy for an expert designer to do but harder for a machine. While the work in [7] eliminated the tediousness of repeatedly laying out component in detail, the entangled complexity of trying to solve semantic layout problem outweighed the benefits. A smaller piece of work in [8] is much more practical and use a similar technique as deployed by our work here of laying out a small number of components piecemeal.

There has been related research on generating user interfaces from task-based models. The work by Paterno in a series of publications [9, 10, 11] focused on a tool for modeling tasks and automatically generating user interfaces from high-level specification and from analyzing task structures. The task model is device independent therefore can automatically generate to multiple platforms [10]. Venderdonkt in [12] does semi-automatic generations through Automatic Interaction Objects structure. Our work uses the business task model as the input into the design process. We are using business task as requirements but currently are not using it to the full extent in helping generating the UI. We find that large enterprises have rather complex workflow and work context, and only small portion of this complexity is captured in business process model. User-centered designers still depend a great deal on closing the knowledge gap through business requirements and user observations to be able to properly complete their work.

Last but not least, a company called iRise [13] of which products center around simulating UI design and visualizing IT solutions early in the design phase. Their products are capable of generating rather attractive UI. Our work shared a common goal of the ability to visualize IT solution early. However, our work aims also at transporting the low-fidelity mock-ups into high-fidelity as well. Our low-fidelity mock-ups are executable user interface and can be beyond presentation materials.

3 Our Approach

In this section, we will walk the readers through our end-to-end approach. We start from a business model (Section 2.1) then transform it into various views in our user-centered the Human Interactions (HI) perspective (Section 2.2). These views are designed to provide different aspects of user-centric information harvested from a business process. The views are also designed as places upon which a design such as pages and grouping of information are built. Section 2.3 describes how design pages are roughly laid out and filled with some automation help. Section 2.4 describes the user interface generation and illustrates the outputs.

3.1 Business Model as a Starting Point

Our starting point is, but not necessarily always, a business process model. Though there are several process notations used in the industry [14, 15], we find the basic information in business processes having to do with “who” will do which “task” “when” in the business process, the sequence of business tasks, and what are the inputs and outputs of a task, to be the most useful set of information for starting a user interface design. This set of information interestingly shared some characteristics with the user task model commonly used as a starting point for user-centered design process [16]. Let’s put aside for later, if it occurs in your mind, whether business modeling and user task modeling, if done in the same tools, would yield the same “task model.” In our approach, we gladly take the business process as is and turn it into perspectives that are more in tune with the user-centered design thinking. Before we embark into further detail, let’s look at an example of an invoice-to-cash business process in Figure 1. Please note here that the complete invoice-to-cash process is rather large consisting of several process diagrams. We only reveal a snippet of the process to illustrate the examples in this paper. For confidentiality reasons, we will not reveal much further information on the entire model.

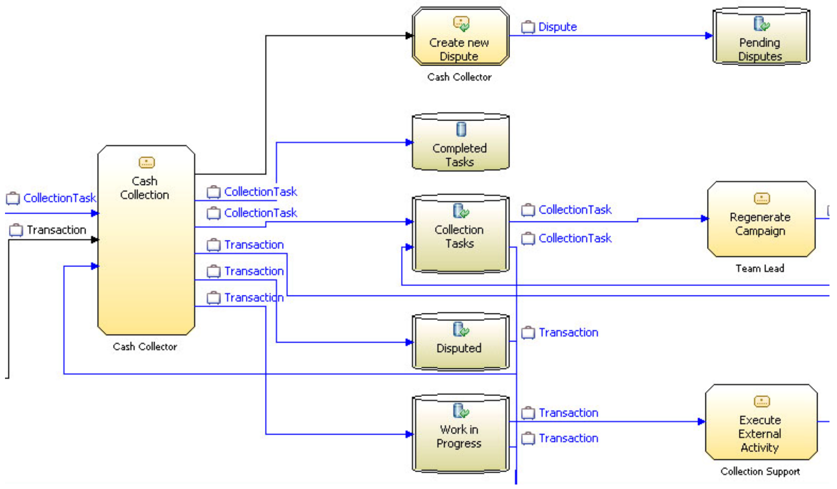


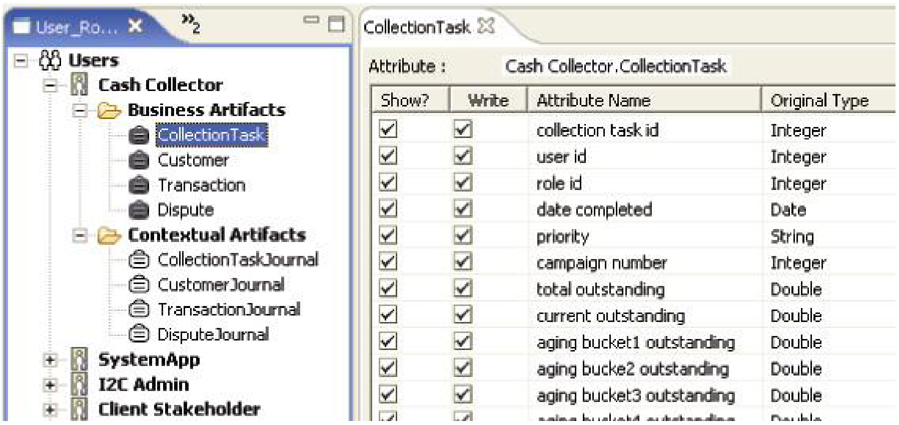
Fig. 1. A snippet of the Invoice to Cash business process shown in the Process Editor in IBM Websphere Business Modeler 6.0

From Figure 1, we want to point out information that is essential for understanding the following steps in our approach. Notice the "Cash Collection" task at the leftmost of the diagram. A user role "Cash Collector" is identified below the task. The cash collection task, as we learned from SMEs and user observations, typically involves the user calling customers based on "transactions," which is one type of input to the task. Another type of input is "collection task" – a group of transactions recommended to Cash Collector for collection. Documents that flow between tasks are referred to in this paper as "information artifacts." Attributes of each artifact in this model and their data types can be defined through other editors in WBM. The directions of arrows indicate the directions of the process flow. Cylinders denote repositories in which information artifacts in various stages are stored.

3.2 Transformation into Human Interaction Perspective

The next step in our approach is converting the business process model into the HI perspective to give designers a schematic summary of the process. There are 4 views in the HI perspective:

- "User role – Artifacts" – shows all information artifacts needed per user role
- "Artifacts – User role" – shows per each artifact user roles with access to it
- "User role – Task" – shows business tasks required for each user role
- "User role – Design"– shows UI design pages that have been defined for each user role. This view has empty content initially.



- Users
  - Cash Collector
    - Business Artifacts
      - CollectionTask
      - Customer
      - Transaction
      - Dispute
    - Contextual Artifacts
      - CollectionTaskJournal
      - CustomerJournal
      - TransactionJournal
      - DisputeJournal
  - SystemApp
  - I2C Admin
  - Client Stakeholder

Attribute : Cash Collector.CollectionTask

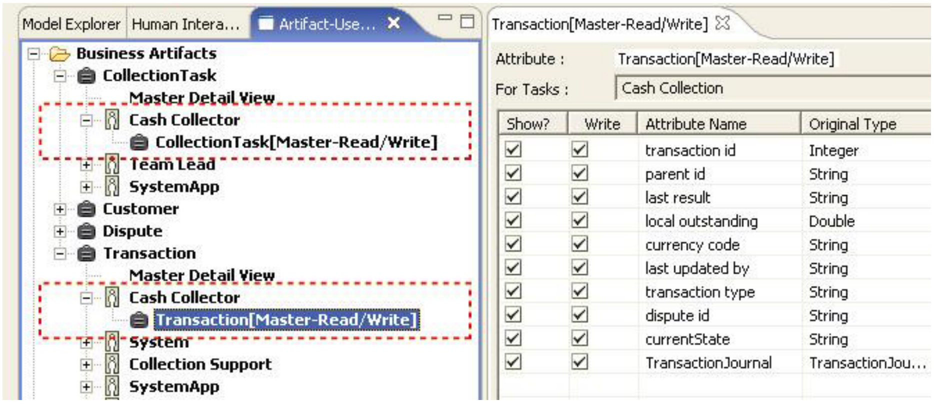
Show?	Write	Attribute Name	Original Type
✓	✓	collection task id	Integer
✓	✓	user id	Integer
✓	✓	role id	Integer
✓	✓	date completed	Date
✓	✓	priority	String
✓	✓	campaign number	Integer
✓	✓	total outstanding	Double
✓	✓	current outstanding	Double
✓	✓	aging bucket1 outstanding	Double
✓	✓	aging bucke2 outstanding	Double
✓	✓	aging bucket3 outstanding	Double

Fig. 2. The “User role – Artifact” View

Figure 2 illustrates the “User role – Artifact” view as initially populated from the business model illustrated in Figure 1. The view centers on user roles; it shows all the artifacts that each role requires for performing their tasks extracted from inputs to each business task. From the design point of view, this summarizes part of the design scope – how many user roles am I designing for? It also gives a sub-scope for each user – what type of information, from business perspectives, is needed by each user role. Since our tool is implemented on an Eclipse-based platform, we use a tree-like structure to present the hierarchical relationships of user roles and artifacts in the Eclipse style. The underlying information model that supports the 4 views is not quite so hierarchical but rather with many interconnections. We made efforts to conceal the UML2 model behind the scene and at the same time conform to standard views in the Eclipse environment.

In Figure 2, notice the two types of artifacts. The “Business Artifact” denotes information artifacts that flow from one task to another. The “Contextual Artifact” denotes artifacts that are inputs or outputs to but do not flow across business tasks. We use this information to guesstimate the importance of an artifact. By assuming that if an artifact flows from one task to the next, and often from one user role to another, it must be significant. This is shared information and is often designed with consistency among user roles. Any artifact that does not flow must be contextual and is used only in a particular task context. The model for the two categories of artifacts are actually similar, the separation is purely to provide designers with a cue on which artifacts might need more careful treatment across user roles. When selecting an artifact from the tree on the left, the attributes of the artifact is shown in an attribute editor shown on the right in Figure 2. We will discuss the use of the attribute editor with the next view.

Complimentary to the “User role – Artifact” view is the “Artifact – User Role” view illustrated in Figure 3. In this view, each artifact is shown with a list of user roles that require access to it. The view is designed to help designers with a concrete understanding of shared artifacts. Knowing that an artifact is used by multiple user roles, designers may and often opt to design similar interfaces to the same artifact, unless dictated otherwise by requirements or by their knowledge of the user tasks.



```
Business Artifacts
  CollectionTask
    Master Detail View
    Cash Collector
      CollectionTask[Master-Read/Write]
    Team Lead
    SystemApp
  Customer
  Dispute
  Transaction
    Master Detail View
    Cash Collector
      Transaction[Master-Read/Write]
    System
    Collection Support
    SystemApp
```

Transaction[Master-Read/Write]

Attribute : Transaction[Master-Read/Write]

For Tasks : Cash Collection

Show?	Write	Attribute Name	Original Type
✓	✓	transaction id	Integer
✓	✓	parent id	String
✓	✓	last result	String
✓	✓	local outstanding	Double
✓	✓	currency code	String
✓	✓	last updated by	String
✓	✓	transaction type	String
✓	✓	dispute id	String
✓	✓	currentState	String
✓	✓	TransactionJournal	TransactionJou...

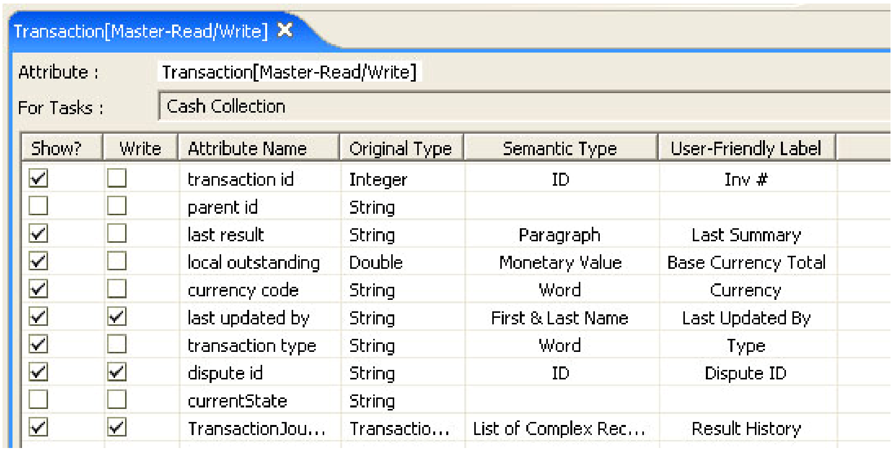
Fig. 3. The “Artifact – User Role” view

The “Artifact – User Role” view is designed to expose business requirements on read/write access to artifacts. This is derived from the business model; if an artifact flows in and out of a business task, we assume the user must update it in the task. Notice in Figure 3 that the Cash Collector has “read/write” access to the “Transaction” and “Collection Tasks” artifacts. For example, the “Cash Collector” user role has a read/write “master” copy of the “Transaction” artifact. Any information refined for the UI in this master copy will govern all partitions of this artifact when they apply to UI designs. This statement will be made clearer in Section 2.3 with examples of “data groups,” subsets of this master copy organized to feed into various pages of the UI design.

A user role may have “read/write” and/or “read only” master copies of an artifact. Read/write means the user can modify the information while “read only” means the user can only view it. When a user role has both “read/write” and “read only” access, it means the user cannot always update the information. A common use case is, for example, a contract may not be modified after it is signed. Being explicit about read/write access brings attention to the designer on how the UI may show information artifacts in various stages of the process.

When a master copy is viewed in the attribute editor, the tasks that are applicable to the access rights are shown in the “For Tasks:” attribute illustrated in Figure 4. Since read/write access is derived at the artifact level from the business process, through the attribute editor, the designer can enrich the information by being more specific – whether a user role needs to see all or part of data elements of an artifact, and whether the user can update all or part of data elements of the artifact. This indication of access rights plays an important part of our tool’s automation. We have an automatic generation capability that uses data elements and their read/write access to initially call out appropriate UI elements.

The attribute editor also provides the designer with an opportunity to put UI related information with data elements of an artifact to be used throughout her design. She can place a user-friendly label for each data element. In this editor, we introduce a notion of “UI semantic type” which better describes the UI semantics of data elements. For example, a basic type string may in deed be a “social security number.” We use UI semantic types to figure out appropriate display format, i.e., a social security number should be displayed as 222-33-4444. We also use semantic types to help pick appropriate sample data for the UI mock-ups. Figure 4 illustrates what the designer may have done in the attribute editor to enrich the “Transaction” artifact.

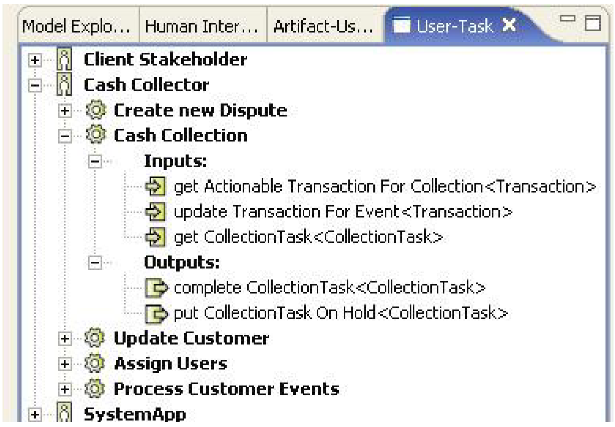

Transaction[Master-Read/Write]

Attribute : Transaction[Master-Read/Write]

For Tasks : Cash Collection

Show?	Write	Attribute Name	Original Type	Semantic Type	User-Friendly Label
✓	☐	transaction id	Integer	ID	Inv #
☐	☐	parent id	String		
✓	☐	last result	String	Paragraph	Last Summary
✓	☐	local outstanding	Double	Monetary Value	Base Currency Total
✓	☐	currency code	String	Word	Currency
✓	✓	last updated by	String	First & Last Name	Last Updated By
✓	☐	transaction type	String	Word	Type
✓	✓	dispute id	String	ID	Dispute ID
☐	☐	currentState	String		
✓	✓	TransactionJou...	Transactio...	List of Complex Rec...	Result History

Fig. 4. Attributes of the Transaction artifact shown in the Attribute Editor after modifications



- Client Stakeholder
- Cash Collector
  - Create new Dispute
  - Cash Collection
    - Inputs:
      - get Actionable Transaction For Collection<Transaction>
      - update Transaction For Event<Transaction>
      - get CollectionTask<CollectionTask>
    - Outputs:
      - complete CollectionTask<CollectionTask>
      - put CollectionTask On Hold<CollectionTask>
  - Update Customer
  - Assign Users
  - Process Customer Events
- SystemApp

Fig. 5. The “User role – Task” view

The “User role – Task” view, illustrated in Figure 5, summarizes all business tasks for each user role. Each business task shows inputs, information needed for the task, and outputs, the outcomes of the task. Using this view, the designers are informed of the governing business tasks for each user and should be able to design appropriately.

We will defer the description of the “User role – Design” view to the next section.

3.3 Building Up the Design with Pages and Data Groups

The designer can now start building up the UI design. Imagine this is when a designer would start sketching out design ideas. Let's assume after some initial thoughts, the designer tries out a couple of pages. Figure 6 shows the skeletons of the 2 pages for the Cash Collector user role. While our visual skeletal sketching tool has not been implemented, here is what the designer might visualize in mind and what they might do had the tool existed.

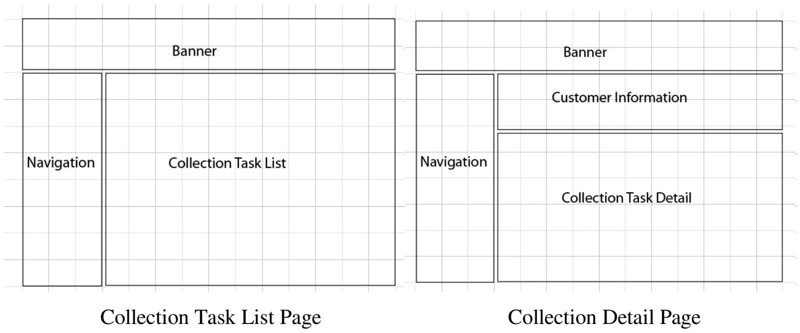
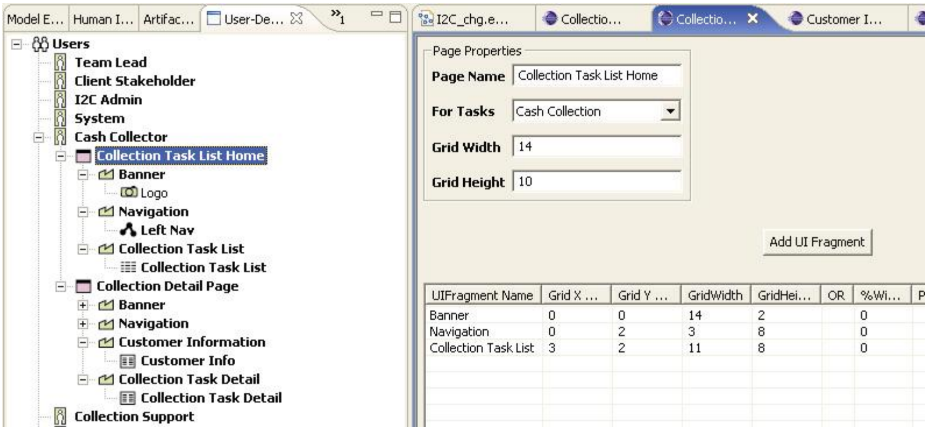


Fig. 6. Visual sketches of Cash Collector interface

The design creates the 2 pages by associating them to the user role Cash Collector and to the "Cash Collection" task. Figure 7 illustrates the page structure above in the "User role – Design" view. In this view, pages associated to a particular user role are specific to that user role. Pages associated with "users" at the top of the user tree are designed to be reused among multiple user roles. A page consists of "UI fragments," which represent visual blocks as shown in Figure 6. A UI fragment is a portion of a page that displays related UI elements, etc. Partitioning a page into UI fragments is purely the designer's discretion.

Next the designer focuses on the information to be placed in various UI fragments on the pages. Here we introduce a "data group" notion denoting a group of selected attributes from an artifact's master copy. A data group singles out data elements chosen to be displayed in a UI fragment. For example, the collection task list in the first page will show a list of tasks, each is shown with 5 elements; the list will contain 5 columns, each column corresponds to a data element in the Collection Task artifact. Figure 8 illustrated the data group for such a list. Notice that in a data group, the designer has another opportunity to localize UI information, for example, the user-friendly label may change to fit a small design space. (In this example, the customer "Location" is re-labeled "Loc" so it occupies a narrower column in the table.) The resulting UI is shown in Figure 9. Read-write access can be reduced to read-only as it is more appropriate for a tabular display in this example. Notice that changing from read only to read/write is not allowed as doing so would violate the restrictions from the business model. For the data group in Figure 8, the sequence of the selected data elements is also shuffled to control the column sequence; this is the way the designer can control the design output.



```
Users
  Team Lead
  Client Stakeholder
  I2C Admin
  System
  Cash Collector
    Collection Task List Home
      Banner
        Logo
      Navigation
        Left Nav
      Collection Task List
        Collection Task List
    Collection Detail Page
      Banner
      Navigation
      Customer Information
        Customer Info
      Collection Task Detail
        Collection Task Detail
  Collection Support
```

Page Properties

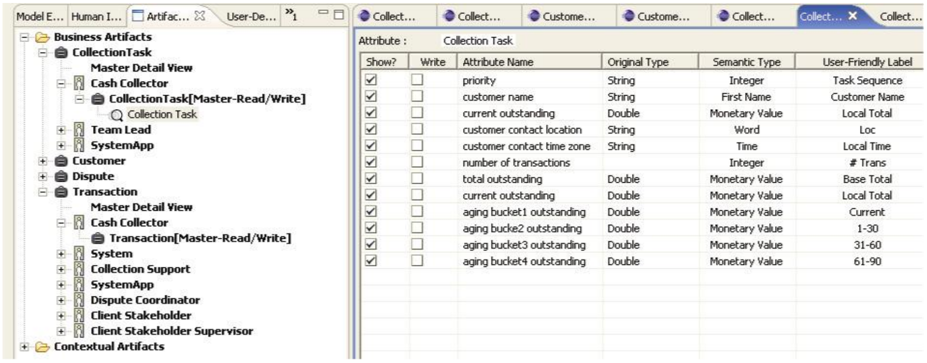
Page Name	Collection Task List Home
For Tasks	Cash Collection
Grid Width	14
Grid Height	10

UIFragment Name	Grid X ...	Grid Y ...	GridWidth	GridHei...	OR	%Wi...
Banner	0	0	14	2		0
Navigation	0	2	3	8		0
Collection Task List	3	2	11	8		0

Fig. 7. Page structure defined in the "User role – Design" view

Note that designers may add data elements had they found through UCD investigations that some additional information is needed by the users. Data elements can be added to the "Master" copy of an artifact hence the addition can be used across the board and can be included in various data groups. Contextual artifacts can also be added had the designer found artifacts undocumented in the business model that the user needs to support a task. When data elements or artifacts are added, discussions need to happen with business analysts and architects of how appropriate and plausible to support such information. The result is often a compromise.

We do not allow business artifacts to be added for obvious reasons.



```
Business Artifacts
  CollectionTask
    Master Detail View
    Cash Collector
      CollectionTask[Master-Read/Write]
        Collection Task
    Team Lead
    SystemApp
  Customer
  Dispute
  Transaction
    Master Detail View
    Cash Collector
      Transaction[Master-Read/Write]
    System
    Collection Support
    SystemApp
    Dispute Coordinator
    Client Stakeholder
    Client Stakeholder Supervisor
Contextual Artifacts
```

Attribute : Collection Task

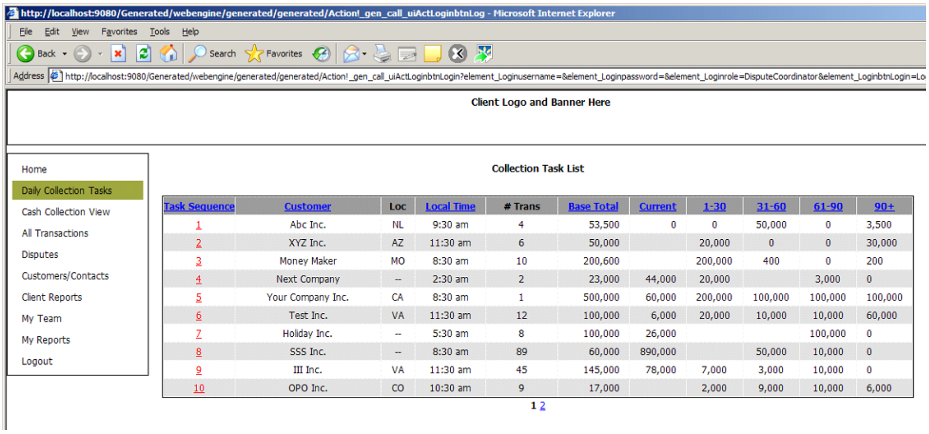
Show?	Write	Attribute Name	Original Type	Semantic Type	User-Friendly Label
✓	☐	priority	String	Integer	Task Sequence
✓	☐	customer name	String	First Name	Customer Name
✓	☐	current outstanding	Double	Monetary Value	Local Total
✓	☐	customer contact location	String	Word	Loc
✓	☐	customer contact time zone	String	Time	Local Time
✓	☐	number of transactions		Integer	# Trans
✓	☐	total outstanding	Double	Monetary Value	Base Total
✓	☐	current outstanding	Double	Monetary Value	Local Total
✓	☐	aging bucket1 outstanding	Double	Monetary Value	Current
✓	☐	aging bucke2 outstanding	Double	Monetary Value	1-30
✓	☐	aging bucket3 outstanding	Double	Monetary Value	31-60
✓	☐	aging bucket4 outstanding	Double	Monetary Value	61-90

Fig. 8. A data group showing selected attributes for listing Collection Task artifacts

A page is not completed unless the content is filled. In our tool, the designer may choose to use high-level components such as a tabular list or a data layout component and associate data groups to them. The data layout component is one of the most useful components and can be specified to layout matching UI elements in 1 or multiple columns. Additional UI components such as a button, a label, etc., can also be manually added to a UI fragment. Due to space limitation, we will omit detail on this aspect of the tool.

3.4 Generating the User Interface from the Model

Once pages are sketched out with data groups associated to the appropriate UI components, designers can generate a user interface to see the results. In addition to the page layout, the design can also define how pages are linked (not discussed in this paper.) To generate a UI, the design model is transformed to the execution-oriented XML model, which is used to further generate the UI pages and connections to a simulated backend system. Figures 9 and 10 illustrate the resulting interfaces of the 2 pages mentioned in the previous section.

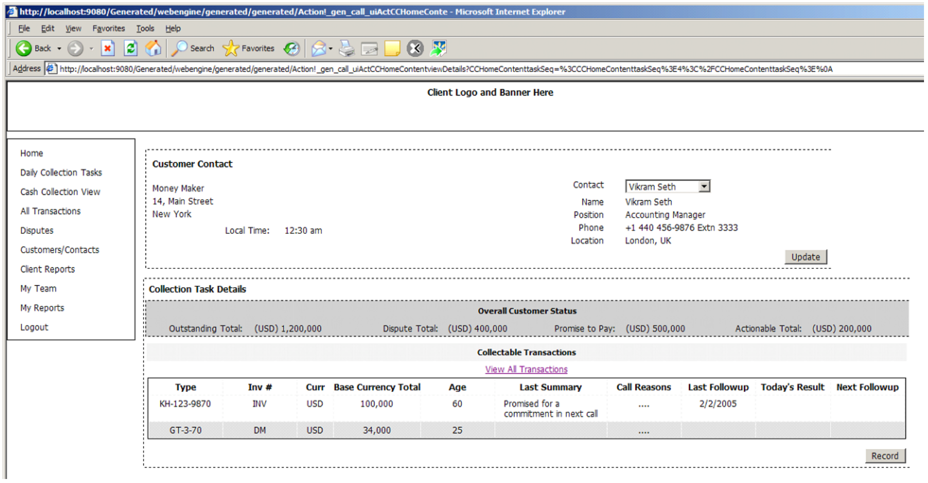

http://localhost:9080/Generated/webengine/generated/generated/Action!_gen_call_uiActLoginbtnLog - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:9080/Generated/webengine/generated/generated/Action!_gen_call_uiActLoginbtnLogin?element_Loginusername=&element_Loginpassword=&element_Loginrole=DisputeCoordinator&element_LoginbtnLogin=Lo
Client Logo and Banner Here
Home
Daily Collection Tasks
Cash Collection View
All Transactions
Disputes
Customers/Contacts
Client Reports
My Team
My Reports
Logout

Collection Task List

Task Sequence	Customer	Loc	Local Time	# Trans	Base Total	Current	1-30	31-60	61-90	90+
1	Abc Inc.	NL	9:30 am	4	53,500	0	0	50,000	0	3,500
2	XYZ Inc.	AZ	11:30 am	6	50,000		20,000	0	0	30,000
3	Money Maker	MO	8:30 am	10	200,600		200,000	400	0	200
4	Next Company	--	2:30 am	2	23,000	44,000	20,000		3,000	0
5	Your Company Inc.	CA	8:30 am	1	500,000	60,000	200,000	100,000	100,000	100,000
6	Test Inc.	VA	11:30 am	12	100,000	6,000	20,000	10,000	10,000	60,000
7	Holiday Inc.	--	5:30 am	8	100,000	26,000			100,000	0
8	SSS Inc.	--	8:30 am	89	60,000	890,000		50,000	10,000	0
9	III Inc.	VA	11:30 am	45	145,000	78,000	7,000	3,000	10,000	0
10	OPO Inc.	CO	10:30 am	9	17,000		2,000	9,000	10,000	6,000

1 2

Fig. 9. The generated UI for the collection task list page


http://localhost:9080/Generated/webengine/generated/generated/Action!_gen_call_uiActCCHomeConte - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:9080/Generated/webengine/generated/generated/Action!_gen_call_uiActCCHomeContentviewDetails?CCHomeContenttaskSeq=%3CCCHomeContenttaskSeq%3E4%3C%2FCCHomeContenttaskSeq%3E%0A
Client Logo and Banner Here
Home
Daily Collection Tasks
Cash Collection View
All Transactions
Disputes
Customers/Contacts
Client Reports
My Team
My Reports
Logout
Update
View All Transactions
Record

Customer Contact

Money Maker
14, Main Street
New York

Local Time: 12:30 am

Contact	Vikram Seth
Name	Vikram Seth
Position	Accounting Manager
Phone	+1 440 456-9876 Extn 3333
Location	London, UK

Collection Task Details

Overall Customer Status

Outstanding Total: (USD) 1,200,000 Dispute Total: (USD) 400,000 Promise to Pay: (USD) 500,000 Actionable Total: (USD) 200,000

Collectable Transactions

Type	Inv #	Curr	Base Currency Total	Age	Last Summary	Call Reasons	Last Followup	Today's Result	Next Followup
KH-123-9870	INV	USD	100,000	60	Promised for a commitment in next call	....	2/2/2005		
GT-3-70	DM	USD	34,000	25		....			

Fig. 10. The generated UI for the collection detail page

For each page, the XML model contains 4 main types of information:

- Page content: describes the page layout and placements of UI elements.
- Page bindings: lists the bindings from UI elements to the data elements
- Page interactions: defines actions and page transitions
- Styles: low fidelity design assumes default styles. More elaborate styles can be used when extending the model to generate high fidelity UI

We follow the consumer-provider pattern to link the UI to the backend system that provides data. The page content is used to generate the consumer code and page bindings and interactions define the provider. The generated provider mainly consists of stubs to connect to the backend which can provide actual data and other functions through any of the following: web services, simple java classes, RMI etc. In the absence of connections to actual backend systems, the default provider gives dummy data for different UI elements. The dummy data is generated based on the semantic types associated with the attributes in the artifact model. We use a test database which contains sample data for different semantic types.

The Human Interaction (HI) model is captured in both UML2 and XML – UML mostly for supporting the design tool and XML for generating executable UIs. To generate a UI, we transform our generic XML model into a platform-specific XML for IBM Websphere Portlet Factory, a commercial product. The result UI runs on any application server for JSP pages, or as portlets in Websphere Process Server.

The same model can be extended to generate the high fidelity UI and backend connections to the actual web services in a production setting. This would require inputs from the designers for the style information and input from the developers on what information is passed from one page to another or from UI to backend for different events. We are currently looking further into supporting this part of the design process.

4 The Model

The HI model consists of elements that captures information around user roles. Our modeling concerns cover both the support for the UI design thinking for designers as much as capturing the actual content of the visual design, navigation, and user interactions. Many modeling elements are introduced mainly to provide design analysis perspectives, flexibility in the design, and design management. A number of elements, many of which are well hidden, provide technical information to produce support for execution. Following is the list of key concepts and relations in the HI model:

- **User Role:** *a category of targeted end users*
- **Business Artifact:** *semantically related group of information that flows between business tasks*
- **Contextual Artifact:** *semantically related group of information that is an input of a task*
- **Data Element:** *properties of an artifact*
- **Data Group:** *selected data elements of one or more artifacts semantically grouped together for the purpose of displaying related information*
- **Page:** *a set of design equivalent to a full screen*

- **UI Fragment:** *a portion of a page that is used for displaying related information and functions*
- **UI Element:** *type of UI components rendered on the screen. Basic types include text, text input, text area, link, drop down list, radio group, check box group, two way select list, calendar, etc. Complex types include tabs, list table, data layout component, navigation component, etc.*
- **User role – Artifact:** *a relationship showing user roles that need access to an artifact, an artifact that is used by a user role*
- **Artifact – Attribute:** *a relationship between an artifact and its properties*
- **Data Group – Attribute:** *a relationship collecting selected data elements*
- **User Role – Page:** *A relationship between a user role and pages that are designed for the role*
- **Page – UI Fragment:** *A relationship between a page and its UI fragments. Each UI fragment in turn contains information on its height, width, and position on the page.*
- **UI Element – Data Group:** *A relationship between data group and a (composite) UI element. The type of UI element determines how the data group may dictate the automatic selection of detailed UI elements that fill out the fragment.*

5 Experience and Analysis

Using business models as a starting point for the UI design process is an interesting endeavor. One observation we had was that some business processes provide a rather concrete set of business tasks, to the point that designers can almost visualize what happens on the screen just from viewing a business process model. Some business processes are rather abstract. In this case, designers need to do a great deal of research to understand the nature of these abstract business tasks. The example given in this paper is an example of the latter. As business processes are not and will most likely never be user task models, we take business models merely as requirements. Composing user tasks for these business tasks are upon the discretion of the designers.

Our HI perspective and tool are designed such that designers can start the UI design process with or without a business process. They are also designed such that information can be added, specifically user roles, artifacts, and attributes. There are many reasons why this is a viable design. One, business models may not be available at the time designers need to start. Two, there may be yet ambiguous knowledge at the start of the design but user-centered designer may have higher confidence in the accuracy of information they have collected through user observations and could indeed start mocking up design ideas. Third, we observed that there are often additional user roles not called upon in the business model. These roles are often in managerial positions and are not contributing to the operations of the model; they function as overseers and often require reporting and monitoring capabilities in the UI design. Currently we allow adding user roles but supporting monitoring types of interaction designs has not been addressed in the current tooling.

Our approach currently appears very top down. The UI design process is all – bottom up, top down, and sometimes middle out. We are looking to further support the design process by allowing designers to flexibly approach a design.

Our work is still on-going. One obvious extension is the visual skeletal sketching tool and an ability to drag and drop data groups into UI fragments. There are many other features that we have not addressed in the current tool, for example, shared templates in the design hierarchy; appropriate style sheets for low fidelity mock up, to name a few. We plan to take better advantage of business models even deeper such as using the business flow to help start a UI flow, and using the business task tree as a place to create user subtasks, etc. We have not validated the tool with the general user target at this point. Our near future work also includes user validation and more business cases to refine the tool and research concepts.

References

1. Landay, J., Myers, B.: Sketching Interfaces: Toward More Human Interface Design. IEEE Computer, 55–64 (March 2001)
2. Lin, J., Thomsen, M., Landay, J.A: Interactive Design: A Visual Language for Sketching Large and Complex Interactive Designs. In: Proceedings CHI 2002: ACM Conference on Human Factors in Computing Systems, ACM Press, New York (2002)
3. Campos, P., Nunes, N.J: Practitioner Tools and Workstyles for User-Interface Design. IEEE Software 24(1), 73–80 (2007)
4. Wong, Y.Y.: Rough and Ready Prototypes: Lesson from Graphic Design. In: Proceedings of CHI 92: ACM Conference on Human Factors in Computing Systems, pp. 83–84. ACM Press, New York (1992)
5. Campos, P.F., Nunes, N.J.: CanonSketch: a User-Centered Tool for Canonical Abstract Prototyping. In: Bastide, R., Palanque, P., Roth, J. (eds.) EHCI-DSVIS 2004. LNCS, vol. 3425, Springer, Heidelberg (2005)
6. Constantine, L.: Canonical Abstract Prototypes for Abstract Visual and Interaction Design. In: Jorge, J.A., Jardim Nunes, N., Falcão e Cunha, J. (eds.) DSV-IS 2003. LNCS, vol. 2844, Springer, Heidelberg (2003)
7. Kim, W.C., Foley, J.D.: Providing High-level Control and Expert Assistance in the User Interface Presentation Design. In: Proceedings of CHI 93: ACM Conference on Human Factors in Computing Systems, pp. 430–437. ACM Press, New York (1993)
8. de Baar, D., Foley, J.D., Mullet, K.E.: Coupling Application Design and User Interface Design. In: Proceedings of CHI 92: ACM Conference on Human Factors in Computing Systems, pp. 259–266. ACM Press, New York (1992)
9. Paterno, F., Mancini, C.: Model-based Design of Interactive Applications. ACM Intelligence Magazine Winter, 26–37 (2000)
10. Paterno, F., Santoro, C.: One Model, many Interfaces. In: CADUI 2002. Proceedings of 4th International Conference on Computer-Aided Design of user Interfaces, pp. 143–154 (2002)
11. Paterno, F.: Tools for Task Modeling: Where We are, Where We are Headed. In: Proceedings of International Workshop on TAsk MOdels and DIAgrams for user interface design, pp. 10–17 (2002)
12. Bouillon, L., Vanderdonckt, J., Chow, K.C.: Flexible Re-engineering of Web Sties. In: Proceedings of ACM Conference on Intelligent Interfaces, pp. 132–139. ACM Press, New York (2004)
13. http://www.irise.com/
14. Business Process Modeling Notation, http://en.wikipedia.org/wiki/BPMN
15. Aalst, W.V.D., Hee, K.V.: Workflow Management: Models, Methods, and Systems, 2nd edn. MIT Press, Cambridge (2004)
16. Mayhew, D.: The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design, 1st edn. Morgan Kauffmann, San Francisco (1999)