Machine-Assisted Design of Business Processes Using Descriptor Space Analysis

Maya Lincoln, Technion – Israel Institute of Technology
Avigdor Gal, Technion – Israel Institute of Technology
Mati Golani, Ort Braude College
Business Process Repositories describe the “know-how” of organizations.

Business Process Repositories can be used for:
- Management of regulations and compliance enforcement
- Management and control of IT systems
- Analysis and improvement of processes
- Documentation and training
- Mergers and acquisitions planning
- Performance monitoring
Motivation

- Process modeling is considered a manual, labor intensive task
  - The outcome depends on personal domain expertise
  - Errors or inconsistencies can lead to bad process performance and high process costs
- Hence, automating the reuse of constructs, gathered from predefined process models does not only save design time but also supports non-expert designers in creating new business process models
Motivation /2 – An Example

- Consider an airport process model that incorporates processes related to passengers check-in before boarding an airplane.
- Now, suppose that the airport management desires to extend the services provided to its customers by offering a new service: “check-in from home”.
- In addition, it is also desired to outline the “check-out” process model as an extension of the current repository.
- The existing repository encapsulates know-how and business logic that are relevant and useful for the creation of these new models.
  - e.g. passenger check-in policies and procedures regarding security, luggage handling, passenger handling and document validation.
In the above scenario, it would have been helpful for the process designer to design the new processes using a supporting system that relies on the reuse of previous know-how instead of creating the model manually from scratch.

To illustrate our methodology we use a real-world case study for airport process design.

Based on a “check-in” process that already exists in the repository, we demonstrate how it is possible to design the two, above mentioned, new business processes.
Research objective

- Propose an effective method for designing new business process models related to any functional domain, without limiting the focus to a specified functional area.
- Delineate new business process models according to the organization’s specific business logics and business rules.
Related work

- Most previous work focused on supporting the design of alternative process steps within existing process models.
- Less work has been carried out on the design of new process models.
- The few works that addressed the design of new models were limited to a specific domain such as production management.
An example: the passenger check-in process

The process descriptor model
A Descriptor Model for Process Design

Object taxonomies

- An object hierarchy model

- An object lifecycle model
A Descriptor Model for Process Design

Action taxonomies

- An action hierarchy model

- An action lifecycle model
The Descriptor Space - Definition

- A quad-dimensional space of activities
  - Each space coordinate represents an activity as a quadruple \( AC = <O,OQ,A,AQ> \)
  - Some coordinates represent “real” activities from the process repository, while others represent “virtual” activities

- The distance between every two coordinates
  \[
  Dist(AC_i, AC_j) = OD_{ij} + AD_{ij} + OHD_{ij} + AHD_{ij}
  \]
  - \( OD_{ij} \) – the object distance: the minimal number of steps connecting \( O_i \) and \( O_j \) in the object lifecycle model
  - \( AD_{ij} \) - the action distance: the minimal number of steps connecting \( A_i \) and \( A_j \) in the action sequence model
  - \( OHD_{ij} \) - the object hierarchy distance: the minimal number of steps connecting \( O_i \) with \( O_j \) in the object hierarchy model
  - \( AHD_{ij} \) – the action hierarchy distance, defined similarly to \( OHD_{ij} \)
  - A “no-connection” distance is used when \( OD/AD \) are undefined
The Descriptor Space – An Example for Calculating Distances

- Consider the two descriptors:
  - $AC_i = (\text{luggage}, \text{hand}, \text{check}, \text{null})$ and
  - $AC_j = (\text{luggage}, \text{null}, \text{get}, \text{from the conveyer belt})$

- To navigate from $AC_i$ to $AC_j$:
  - We move one step up in the object hierarchy ($OHD = 1$) from the object Hand luggage to the object Luggage
  - Then, we recede two steps back from the action Check in the action sequence ($AD = 2$), resulting with the action “Get”
  - Finally, we drill down one step within the action hierarchy ($AHD = 1$), and retrieve the action “Get” from the conveyer belt, and by that we reach the target descriptor
  - The total distance between the two above coordinates is 4
The Process Navigator
The Process Navigator /2

- Suggesting the First Process Activity
  - Goal
  - Search the target object and its more specific objects within the object hierarchy model
  - Match it with an initial action that can be acted on this object
  - Compose first activity suggestions
    - Retrieved objects and the first action that acts upon them
  - Sort and flag results
The Process Navigator

- Refining the Currently Suggested Process Activity (e.g. “Get luggage”)
  - Action and Object Refinement
    - E.g. “Get luggage from the conveyer belt”, “Get hand luggage”
  - Action and Object Generalization
  - Advance an Object's State or an Action
    - The object “Standard luggage” represents a more advanced state of the object “Luggage”
    - The action “Give” follows “Get” in the action sequence applied on “Luggage”
    - => The following refinement suggestion is constructed: “Get standard luggage”, and “Give luggage”
The Process Navigator

- Refining the Currently Suggested Process Activity (continue)
  - Recede to a Less Processed State of the Object or to a Former Action
    - E.g. the action “Present” is acted on “Luggage” before this object is taken (before the action “Get” is applied), hence creating the refinement option: “Present luggage”
  - Move to a Sibling Action or Object
    - E.g. a navigation to sibling actions to “Get” retrieves a list of activities that includes: “Check luggage” and “Take luggage”
    - In the same manner, a search for sibling objects, retrieves a list of activities, that includes: “Get passport” and “Get visa”
The Process Navigator /5

- Suggesting the Next Process Activity
  - Goal: take the process execution flow one step forward
  - Two alternative ways:
    - Advancing to a later action that acts on the currently accepted object
      - E.g. “Give passport” -> “Check passport” / ”Return passport”
    - Proceeding to a sibling object combined with the reference activity's action
      - Rationale: in some process flows the same action is operated on sibling objects in order to fulfill a certain process goal (e.g. Send standard luggage -> Send excess luggage)
      - E.g. “Give passport” -> “Give visa” / ”Give luggage” / ”Give information”
Preparing a Set of Output Options

- Sort by Proximity to the Reference Activity
  - By calculating distances
- Internally Sort by Similarity to Processes in the Repository
  - *No change* - the suggested activity is represented “as is” within the underlying business process repository. No mark
  - *Slight modification* - there is an actual activity in the underlying business process repository containing the same object and action with different qualifiers. Marked with “~”
  - *Major change* - the object and action within the suggested activity were not coupled in any of the activities within the underlying business process repository. Marked with “M”.

- Add a Random Option
- Flag Each Option
  - E.g. “[1,~]”
Implementation

- Server side logic is implemented in PHP using a MySQL database.
- The client runs within an Internet browser and is implemented in HTML and JavaScript, with AJAX calls to the server.
Case Study

- Based on the aviation process repository
- Designing a new process: “Passenger Checkout”
  - Extends the process repository by handling passenger related activities conducted after an airplane arrives at its destination
  - Final design output:
Case Study – process generation system

- Step 1: The process designer’s input
Case Study process generation system

- Step 2: First activity (defined by the designer) is: “Give passport”
- Step 3: Next activity suggestions:
Case Study /4

- Step 4: The designer selects the option “Check passport”
- Step 5: The designer selects the option “Give luggage” as a next future activity (will be required at the customs point)
- Step 6: The designer then asks the process navigator to provide next step options and receives:
- Step 7: The designer selects the first option, “Check luggage”
Case Study

Step 8: The designer asks for previous activity suggestions to “Give luggage”

- Rationale: by reviewing the newly designed process, she realizes that an activity may be missing before Give luggage, since the passenger may not have carried his luggage with him to the airplane.

Step 9: Retrieved previous step suggestions (by navigating backwards in the action sequence)

Step 10: The designer selects the option: “Get luggage” and asks the process navigator to refine it

- Reason: it seems to lack sufficient details to express the activity required in this context
Step 11: The process navigator presents refinement suggestions

Step 12: The designer selects the option: “[1,~] Get luggage from the conveyer belt”

• Note that this activity was selected although it was not represented “as is” in the business process repository
Case Study

- Designing the new process: “Send luggage from home”
- Output:

  ![Process Diagram]

  - An interesting observation is the usage of the activity “Put luggage in vehicle”
    - While the original business process repository contained the action “Put in vehicle” applied only to the object “Baby carriage”, the terminating activity combines this action with the object: “Luggage”
We chose a set of 14 processes from the Oracle Business Model (OBM)
• nine business processes from the Procurement category (96 activities)
• five business processes from the Inventory category (31 activities)

The Procurement data set contains related, sequential activities and therefore encapsulates a focused operational area

The Inventory data set encapsulates a loosely coupled business logic regarding an extended business area
At each experiment, a single process was removed from the database and was reconstructed using the “New Process Design Assistant” software (NPDA). This way, the missing process serves as the final design goal, enabling us to measure the method's effectiveness in an objective manner.

Each experiment was conducted according to the following steps:

- Remove one of the processes from the database so that the database will not contain any of its activities.
- Run the NPDA and select at each phase the option (activity) compatible with the removed process.
  - Handle cases in which no option represents the goal activity.
Experiments - Methodology

- At each experiment, a single process was removed from the database and was reconstructed using the “New Process Design Assistant” software (NPDA).
- This way, the missing process serves as the final design goal, enabling us to measure the method's effectiveness in an objective manner.
- Each experiment was conducted according to the following steps:
  - Remove one of the processes from the database so that the database will not contain any of its activities.
  - Run the NPDA and select at each phase the option (activity) compatible with the removed process.
  - Handle cases in which no option represents the goal activity.
## Experiment Results

<table>
<thead>
<tr>
<th>Column name</th>
<th>Column # 1</th>
<th>Column # 2</th>
<th>Column # 3</th>
<th>Column # 4</th>
<th>Column # 5</th>
<th>Column # 6</th>
<th>Column # 7</th>
</tr>
</thead>
<tbody>
<tr>
<td># of total processes in DB</td>
<td># of total activities in DB</td>
<td>% of goal activities represented in the DB</td>
<td>Avg. # of steps per design phase</td>
<td>Avg. location of correct option in ’next activity’</td>
<td>Avg. location of correct option in ’refine activity’</td>
<td>Avg. location of the correct option per design phase</td>
<td></td>
</tr>
<tr>
<td>Avg.-all</td>
<td>14</td>
<td>127</td>
<td>89.0%</td>
<td>2.0</td>
<td>1.2</td>
<td>2.8</td>
<td>2.6</td>
</tr>
<tr>
<td>Avg.-Procurement</td>
<td>9</td>
<td>96</td>
<td>90.6%</td>
<td>1.9</td>
<td>0.8</td>
<td>3.0</td>
<td>2.8</td>
</tr>
<tr>
<td>Avg.-Inventory</td>
<td>5</td>
<td>31</td>
<td>83.9%</td>
<td>2.1</td>
<td>1.9</td>
<td>2.4</td>
<td>2.3</td>
</tr>
</tbody>
</table>

Table 1. Experiment results.
## Experiment Results

**Table 2.** Distribution of successful predictions vs. the number of required refinements.

<table>
<thead>
<tr>
<th># of refinements</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of successful predictions</td>
<td>12%</td>
<td>35%</td>
<td>27%</td>
<td>12%</td>
<td>4%</td>
<td>2%</td>
<td>2%</td>
<td>1%</td>
<td>1%</td>
<td>3%</td>
</tr>
<tr>
<td>Cumulative</td>
<td>12%</td>
<td>48%</td>
<td>75%</td>
<td>88%</td>
<td>92%</td>
<td>94%</td>
<td>96%</td>
<td>96%</td>
<td>97%</td>
<td>100%</td>
</tr>
</tbody>
</table>
Conclusions

The proposed method, software tool, and experiments provide a starting point that can already be applied in real-life scenarios, yet several research issues remain open, including:

1. an extended empirical study to further examine the quality of newly generated processes;
2. an extended activity decomposition model to include an elaborated set of business data and logic (e.g., roles and resources); and
3. defining a learning mechanism that will take into account previous designer preferences and adjusting (in real time) the process delineator mechanism.
Conclusions/2

- As a future work we intend to investigate further language semantics by using more advanced natural language processing techniques, as well as semantic distances between words.
- Finally, we intend to apply the techniques we have developed to create new methods for workflow validation.
Thank you!