Modelling and analysis of human collaboration processes in organizations

Stuit, Marco

Appendices

Appendix A – Pseudo Code E-mail Interaction Mining Method

The email interaction mining method from Chapter 4 to discover an interaction-centric process model from a set of e-mail messages is defined in the following (tool-independent) activities and steps.

Activity 1: Pre-Processing

1. Determine the organization's project or business process under consideration
2. Identify the involved employees
3. Collect the relevant e-mail messages from the employees' *Inbox* folders
4. Store all messages in a single archive

Intermediate output: a manually pre-processed e-mail archive with messages that pertain to a single process instance

Activity 2: Thread Identification

1. Group all messages by thread
   a. Build simple threads using *threading by subject*
   b. Build compound threads using *threading by reference*
   c. Identify the root and child threads within compound threads using *threading by subject*
2. For each thread, record thread attributes
   a. Set the name to match the subject line of the root e-mail
   b. Set the initiator to match the author of the root e-mail
   c. Set the start time to match the origination date of the chronologically first message in the thread
   d. Set the end time to match the origination date of the chronologically last message in the thread
   e. Set the parent thread to match the name of the parent thread

Intermediate output: a set of e-mail threads and their attributes

Activity 3: Interaction Identification

1. For each simple thread
   a. Create an atomic interaction
   b. Set the attribute *Name* to match the thread name
   c. Set the attribute *Class* to *Atomic*
2. For each compound thread
   If the root thread contains one e-mail message then[36]
   a. Create a complex interaction based on the root thread
   b. Set the attribute *Name* to match the name of the root thread

[36] In this case, all replies in the compound thread introduce a new subject.

c. Set the attribute *Class* to *Complex*
d. For each child thread in the compound thread
   - Create a simple interaction
   - Set the attribute *Name* to match the name of the child thread
   - Set the attribute *Class* to *Simple*
   - Connect the complex interaction as a parent to the simple interaction

If the root thread contains two or more messages then[37]

a. For each thread in the compound thread
   - Create a simple interaction
   - Set the attribute *Name* to match the name of the (root or child) thread
   - Set the attribute *Class* to *Simple*
b. Create an artificial complex interaction
c. Set the attribute *Name* of the artificial complex interaction to match the name of the root thread and attach the string *"[Begin]"*
d. Set the attribute *Class* to *Complex Begin*
e. Connect the artificial complex interaction as parent to the simple interactions

3. Normalize the labels of all the identified interactions by removing common prefixes like "Re:" or "Fwd:"

Intermediate output: (1) a list of *atomic, simple*, *complex*, and *complex begin* interactions, (2) a set of partial interaction structures that represent the parent-child relations between *complex (begin)* and *simple* interactions

Activity 4: Control flow discovery

1. Create an artificial complex interaction to act as the overall root of the interaction model
2. Set the attribute *Class* of the root interaction to *Complex Root*
3. Set the attribute *Routing Type* of the root interaction to *SEQ*
4. Set the attribute *Name* of the root interaction to match the name of the project or business process under consideration
5. Sort all *Complex*, *Complex Begin*, and *Atomic* interactions in ascending chronological order based on their start times
6. Process the interactions from this main list of sorted interactions

   If a direct successor in the list (interaction 2) lies before the end time of a previous interaction in the list (interaction 1) then
   a. Start a parallel chain
   b. Assign interaction 1 as the first member of the parallel chain
   c. Assign interaction 2 as a member of the parallel chain

[37] In this case, there are replies with the subject line of the root e-mail of the root thread.

   d. Assign interactions from the list as members of the parallel chain as long as the start time of those interactions lay before any of the end times of the interactions already assigned to the parallel chain
7. For each parallel chain
   a. Execute **Activity 7**
   b. For each *Complex Parallel* interaction that is created during **Activity 7**
      - Set the attribute *Start time* to match the start time of its chronologically first direct child interaction
      - Set the attribute *End time* to match the end time of its chronologically last direct child interaction
   c. Connect the overall root interaction *Complex Root* as parent to the chain's root interaction
8. Connect the overall root interaction *Complex Root* as parent to the sequential interactions from the main list of step *6*[38]
9. For each *Complex* and *Complex Begin* interaction, list its children in ascending chronological order based on their start times[39]
10. Execute step *7* for each list of *Simple* sibling interactions
    a. Case 1: all the siblings in the list are sequential (i.e. there is no parallel chain)
       - Set the attribute *Routing Type* of their parent interaction to *SEQ*
    b. Case 2: all the siblings in the list are assigned to a single parallel chain
       - Execute step *2* from *Activity 7*
       - Execute step *7b*
       - Set the attribute *Routing Type* of their parent interaction to *PAR*
    c. Case 3: one or more parallel chains are identified with adjacent or interim sequential siblings
       - Set the attribute *Routing Type* of the siblings' parent (interaction 1) to *SEQ*
       - For each interaction that is a member of a parallel chain, remove the parent-child relation with interaction 1;
       - Execute steps *7a* and *7b*
       - Connect interaction 1 as parent to each chain's root interaction
11. Set the attribute *Start time* of the *Complex Root* interaction to match the start time of its chronologically first direct child interaction
12. Set the attribute *End time* of the *Complex Root* interaction to match the end time of its chronologically last direct child interaction

Intermediate output: A temporarily ordered interaction structure with composition and routing relations

[38] These are the interactions that are not assigned as members of the identified parallel chains.
[39] Effectively, this creates multiple lists of sorted Simple sibling interactions.

Activity 5: Attribute assignment

1. For each interaction
   a. Set the attribute *Active participants* to be the unique persons from the "From" and "To" fields of the messages in the corresponding thread
   b. Set the attribute *Passive participants* to be those persons in the "Cc" fields of the messages in the corresponding thread that are not active participants
   c. Set the attribute *Initiator* to match the sender of the root e-mail in the corresponding thread
   d. Set the attribute *Duration* to be the calculated difference between the start- and end times in number of days, hours, and minutes
   e. Set the attribute *Message size* to be equal to the number of messages in the corresponding thread[40]
   f. Set the attribute *Number of children* to match the number of children of the interaction
   g. Set the attribute *Parent* to match the ID of the parent of the interaction

   If the interaction is *Atomic* or *Simple*
   a. Set the attributes *Start time* and *End time* to match the start- and end-times of the corresponding thread

   If the interaction is *Complex*
   a. Set the attribute *Start time* to match the start time of the corresponding root thread
   b. Set the attribute *End time* to match the latest end time of the direct child interactions

   If the interaction is *Complex Root*, *Complex Begin* or *Complex Parallel*
   a. Set the attribute *Start time* to match the start time of the chronologically first direct child interaction
   b. Set the attribute *End time* to match the latest end time of the direct child interactions

Activity 6: Role Identification

1. For all active interaction participants, identify the corresponding role name or job title based on organizational documentation
2. For each *Atomic*, *Simple*, and *Complex* interaction
   a. Connect the identified roles based on the active participants
   b. Connect the initiator role
3. For each *Complex Root*, *Complex Begin*, and *Complex Parallel* interaction
   a. Collect and connect the roles of their direct children
   b. Connect the initiator role of the chronologically first direct child as the initiator role

[40] If there is no e-mail header data, the attributes in *a* to *e* are left empty. This is the case with artificial complex interactions (i.e. Complex Root, Complex Begin, and Complex Parallel interactions).

Output: The output model, an Interaction Structure (IS) diagram including roles

Activity 7: Post-Processing (optional)

1. Perform close reading of the e-mail messages in the archive
   a. Discover missing root or interim messages in existing threads
   b. Discover new connections between existing threads
   c. Discover new embedded threads
2. If any new e-mails or threads are discovered
   a. Modify the interaction structure and the interaction attributes (start time, participants, number of messages) of the corresponding interactions
3. Modify and correct the interaction names using one or more of the following research methods
   a. Inspection of organizational documentation
   b. In-depth interviews with employees
   c. Close reading of e-mail messages

Output: A manually post-processed output model

Activity 8: Determine the control flow structure within the parallel chain

1. For the first interaction in the parallel chain
   a. Create an artificial complex interaction to act as its parent (this is the root interaction of the parallel chain)
   b. Set the attribute *Class* of the artificial complex interaction to *Complex Parallel*
   c. Set the attribute *Name* of the *Complex Parallel* interaction to "[PAR]"
   d. Set the attribute *Routing Type* of the *Complex Parallel* interaction to *PAR*
2. For each subsequent interaction in the parallel chain ask the following questions
   a. Does the interaction completely lie within the time span of its direct predecessor?
      - Yes, go to step 1c
      - No, go to step 1b
   b. Does the interaction lie sequentially with respect to its direct predecessor?
      - Yes, go to step 1c
      - No, add the interaction as a child on the level of its direct predecessor in the tree, and continue on the lowest existing level[41]
   c. Does the interaction have successors that start within its time span?

[41] Each interaction in the IS diagram appears at a certain level in the tree, starting at level zero, which is the root level and highest level.

- Yes, execute the following steps
  - Create an artificial complex interaction to act as its parent
  - Set the attribute *Class* of the artificial complex interaction to *Complex Parallel*
  - Set the attribute *Name* of the *Complex Parallel* interaction to "[PAR]"
  - Set the attribute *Routing Type* of the *Complex Parallel* interaction to *PAR*
  - Connect the *Complex Parallel* interaction as a child to the *Complex Parallel* interaction immediately above it (in the tree)
- No, go to step 1d

d. Does the interaction have predecessors that end within its time span?
   - Yes, add the interaction as a child on the highest predecessor level that ends within its time span, and continue on the lowest existing level
   - No, add the interaction as a child on the level of its direct predecessor, and continue on the lowest existing level

Appendix B – Email Etiquette Rules

Organizations that plan to use the email interaction mining method from Chapter 4 are advised to establish the following e-mail etiquette rules. Adherence to these rules improves the quality (i.e. structure) of the output model and thus its usefulness for analysis and improvement of the email-driven business process under study:

- Create meaningful subject lines for the recipient as well as yourself. Avoid generic subject lines like "Request for information". Instead, use "Request for information about Product X". This ensures that the interaction names in the output model are more informative and appeal to a larger audience;
- Avoid spelling and grammar mistakes in subject lines, and the use of business-specific or technical content/abbreviations. This ensures that the output model can be well understood by a large audience;
- Use the "To" field for recipients who should respond to the message and the "Cc" field for recipients who are only informed. This ensures that the active and passive interaction participants are correctly set;
- Limit messages to one subject, that is, start a new message for a new subject. For instance, start a reply with a new subject line when a thread is diverging. This ensures that the subject is represented as an interaction in the output model since the proposed method does not look into message bodies;
- Click "Reply" instead of "New E-mail" when replying to a message. The latter starts a new thread and incorrectly becomes a separate (unrelated) interaction in the output model;
- Do not use the reply function to save the time and trouble of typing e-mail addresses. This means the new e-mail starts a (otherwise unrelated) child thread, which results in the composition of otherwise unrelated interactions in the output model. An address book is commonly available (or can be easily implemented in popular email clients) to employees that allow them to input real names in recipient fields. In this way, email addresses do not have to be typed;
- Store or archive all messages relating to the same project or process together.

An e-mail policy may be set-up to formally establish these rules in an organization.