Recovering a Business Object Model from Web Applications

U. De Carlini, A.R. Fasolino, P. Tramontana

Dipartimento di Informatica e Sistemistica

University of Naples Federico II, Italy

G.A. Di Lucca

RCOST – Research Centre on Software Technology

University of Sannio, Benevento, Italy

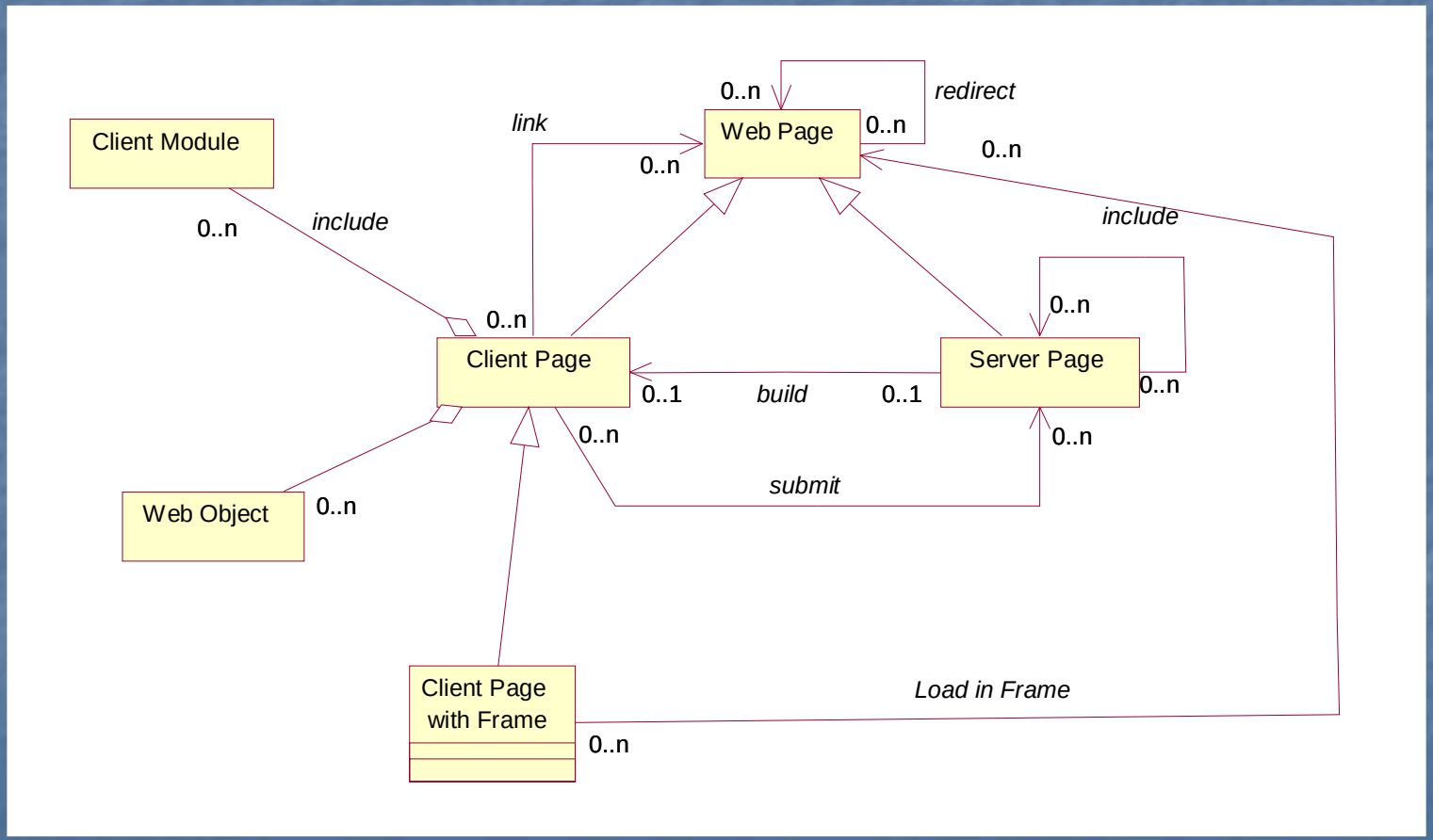
Web Applications (WA): problems and open issues

- World Wide Web is evolving
- WA are realized with a limited quality due to the pressure of a short time-to-market and an extremely high competition
  - WA developed without a disciplined process
  - Poor, inadequate, incomplete documentation
  - Disordered architecture

Web Applications (WA): problems and open issues

- ⇒ Maintenance, Reengineering and Migration are very critical tasks for WA
- ⇒ Object Oriented Business Level Models are a fundamental starting point for these tasks

Web Application Model



Business Level UML Diagrams

- Business level UML diagrams describe the relevant conceptual components (*business objects*) from the domain of the problem addressed by the WA and their mutual relationships and interactions

Recovering a Business Level Class Diagram

A Reverse Engineering process in three main steps:

1) Identification of candidate classes and their attributes;
2) Association of methods to candidate classes;
3) Identification of relationships between candidate classes.

1) Identification of candidate classes

- Searching for groups of logically related data making up the state of objects.

⇒Looking for language mechanisms that allow grouping of related data implementing a relevant concept, either from the domain of the application or from domain of the solution

Identification of relevant groups of data

- Groups of data in input/output operations
  data involved in HTML forms, HTML tables, ...
- Groups of data in database/files read/write operations
  data involved in Recordsets, Arrays, Collections, ...
- Groups of data passed throw distinct page
  data involved in Querystrings

Synonyms and Homonyms Analysis

- *Synonyms* are identifiers with different names but the same meaning
- *Homonyms are* identifiers with the same name but different meanings

⇒*Synonym* identifiers must be assigned with the same unique name.

⇒*Homonym* identifiers must be associated with distinct names.

During *synonyms & homonyms* analysis, a meaningful name is assigned to each data item

An automatic procedure to propose a set of candidate classes

- the identified data groups are arranged in a list and sorted in descending order with the number of references made to each data group, and in ascending order with the cardinality of each group; this sorted list is called OrdList;
- the first group in OrdList is considered as a candidate object and moved into a new list of candidate objects, CAND;
- the OrdList is sequentially visited and each group is considered: if a group comprises at least a new data item not yet included in any other group in the CAND list, it will be inserted in CAND;
- the OrdList is examined until it includes at least a group, or until the union set of all the data items of the candidate objects in CAND and the union set of all the data items of the groups in Glist are equal;
- if a group *h* from the OrdList includes all the data items making up one or more groups *Ci* in CAND, only the *k* data items in *h* that are not yet included in any group of CAND are added to the *Ci* groups whose elements are all included in *h*

2) Associating methods to candidate classes

- Possible functional units to be considered should include web pages, functions, script blocks, depending on the requested degree of granularity.
- The automatic clustering approach proposed in [Di Lucca et al.-IWPC 2002], is used to group Web pages of a WA into meaningful (highly cohesive) and independent (loosely coupled) clusters

$\Rightarrow$Our approach proposes to consider these clusters of related Web pages as potential methods to be associated with the candidate objects of the WA

Coupling between clusters and objects

- Measures of coupling between clusters and objects are computed based on the accesses of each page in the cluster to the candidate object. A page accesses a candidate object when it includes instructions that *define* or *use* the value of some object attribute

Associating pages to candidate classes

⇒ If a cluster accesses exclusively one object, it will be assigned as a method of that object.

⇒ If a cluster accesses more objects, it will be assigned to the candidate class it accesses prevalently

⇒ Clusters that do not make access to any objects will be considered as coordinating modules controlling the executions of other methods

3) Identification of relationships between classes

Relationships are found in two cases:

- If two or more candidate classes have common attributes;
- If a cluster accesses more than one candidate class

Common attributes

- If two or more classes C_i have a common attribute:
  - ⇒ the attribute is assigned to one of these classes (C_j) and deleted from other classes C_k;
  - ⇒ A relationship will be defined between the class C_j and each class C_k

⇒These relationships will be depicted as UML *association* relationships

Clusters accessing more than one class

- ⇒ If a cluster is assigned as a method to a candidate class C_0, but it accesses attributes of other classes C_i, a relationship will be defined between class C_0 and each class C_i
- ⇒Also this kind of relationships will be depicted as UML *association* relationships

A case study

- a WA designed to support the activities of undergraduated courses offered by a Computer Science Department
- Developed with ASP, Javascript, HTML languages
- 75 server pages, 23 client pages, 1 utility module (7648 LOCs)

Identification of groups of data

- After static analysis of the WA:
  - 128 references to data groups (485 data items)
- After Synonyms and Homonyms Analysis:
  - 43 different data groups (26 different data items)

Data Groups

An excerpt of the 43 different Data Groups obtained after Synonyms and Homonyms Analysis

	Data groups	#
C1	Student name, Student surname, Student code, Student email, Student phone number, Student password	14
C2	Teacher name, Teacher surname, Teacher email, Teacher phone number, Teacher password, Teacher code	11
C3	Exam date, Exam time, Exam classroom	10
C4	Student name, Student surname, Student code, Student email, Student phone number	8
C5	Tutoring date, Tutoring start time, Tutoring end time, Course code, Course name	7
C6	Student code	6
C7	Course code, Course name, Course academic year	5
C8	Course code	4
C9	Course code, Course name	4
C10	Teacher name, Teacher surname, Teacher email, Teacher phone number, Teacher password	4
C11	Course code, Course academic year	3

Candidate classes

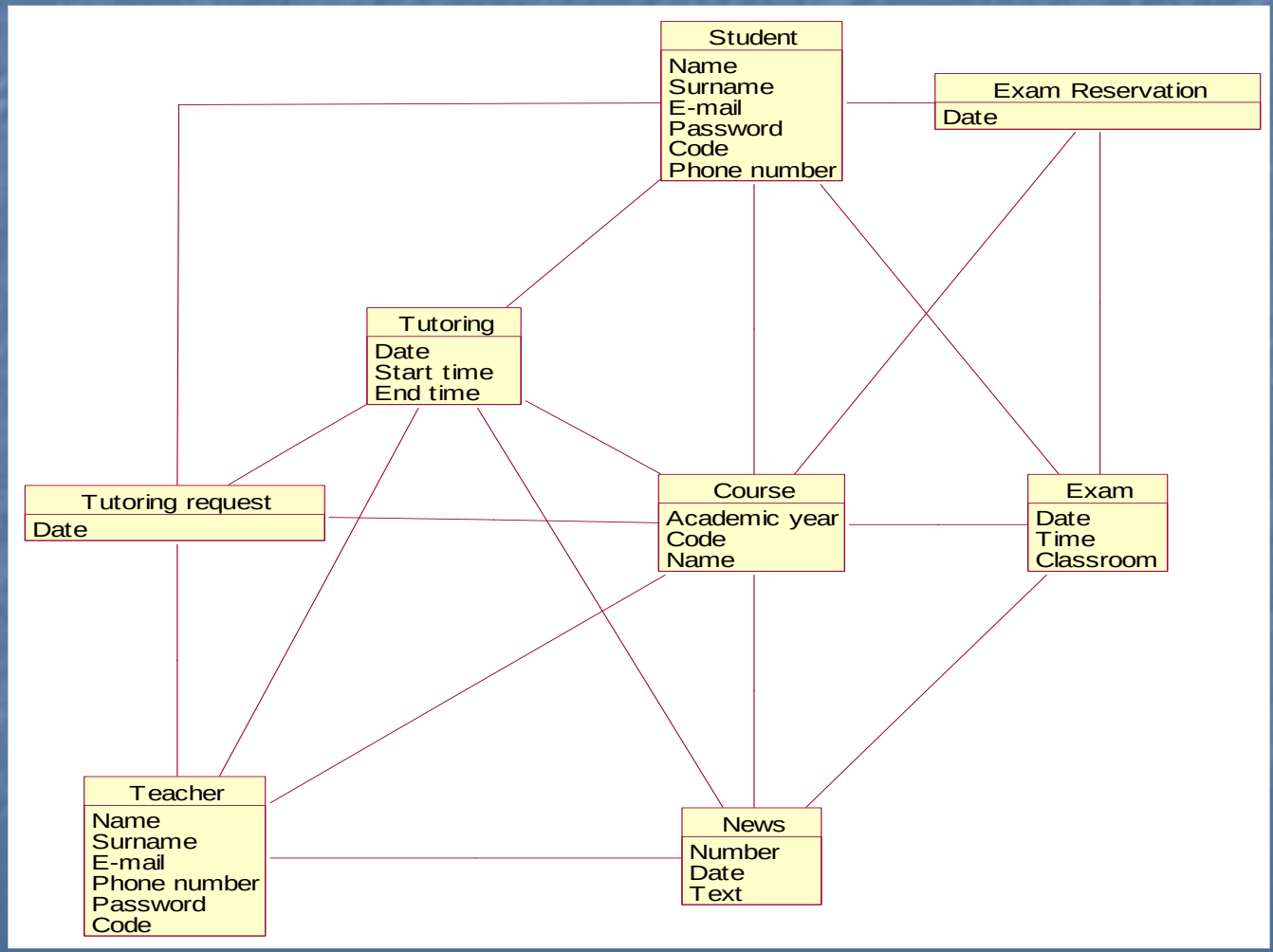
8 Candidate Classes obtained after the execution of the algorithm *Produce Candidate Objects.*

Candidate classes and corresponding attributes	
Student	(Student name, Student surname, Student code, Student email, Student phone number, Student password)
Teacher	(Teacher name, Teacher surname, Teacher email, Teacher phone number, Teacher password, Teacher code)
Exam Session	(Exam date, Exam time, Exam classroom)
Tutoring	(Tutoring date, Tutoring start time, Tutoring end time, Course code, Course name)
Course	(Course code, Course name, Course academic year)
Tutoring Request	(Student name, Student surname, Student code, Tutoring request date)
News	(Course code, News text, News number, News date, Teacher code)
Exam Reservation	(Student code, Student name, Student surname, Course code, Exam date, Exam reservation date)

Associating methods to candidate classes

- Automatic clustering approach was applied
  - 44 valid clusters were recovered
    - 16 clusters did not reference any data group, so they were considered as coordinator modules
    - 28 clusters were assigned as methods of the candidate classes

Business Level UML Class Diagram



Conclusions

- A reverse engineering approaches for recovering, from the code of a Web application, business a Business Object Model has been presented
- Some experiments were carried out to assess the effectiveness of the proposed approaches.
- Encouraging results as to the adequacy of the recovered models were obtained.
- A limited human effort required for reconstructing the models was recorded too.

Future Work

- The definition of criteria for a further automation of the model reconstruction will be addressed, as well as the investigation on possible approaches for identifying UML aggregations, compositions, or generalization-specialization relationships between classes will be carried out.
- A wider experimentation involving more complex Web Applications, implemented with different technologies, will be moreover carried out, in order to extend the validity of the proposed approaches