GeWare 2.0 - On the way to a virtual repository

Toralf Kirsten

01.07.2008
What’s GeWare?

GeWare - Genetic Warehouse

Platform for managing and analyzing large sets of experimental data generated by

- high-throughput expression experiments (Affymetrix)
- custom-made CGH arrays (platform extension)
What’s GeWare?

- GeWare - Genetic Warehouse
- Platform for managing and analyzing large sets of experimental data generated by
  - high-throughput expression experiments (Affymetrix)
  - custom-made CGH arrays (platform extension)
- Generic approach to manage experiment annotations and import clinical data using annotation templates and controlled vocabularies
What’s GeWare?

- **GeWare - Genetic Warehouse**
- **Platform for managing and analyzing large sets of experimental data generated by**
  - high-throughput expression experiments (Affymetrix)
  - custom-made CGH arrays (platform extension)
- **Generic approach to manage experiment annotations and import clinical data using annotation templates and controlled vocabularies**
- **Hybrid integration of web data using SRS**
What’s GeWare?

- GeWare - **Genetic Warehouse**
- Platform for managing and analyzing large sets of experimental data generated by
  - high-throughput expression experiments (Affymetrix)
  - custom-made CGH arrays (platform extension)
- Generic approach to manage experiment annotations and import clinical data using annotation templates and controlled vocabularies
- Hybrid integration of web data using SRS
- Closed loop analysis workflow: Use of chip and gene/clone groups for easy and iterative analysis execution
- Data analysis by R BioCondutor (reuse of existing analysis software)
Content

- What's GeWare
- Requirements
- GeWare 2.0 Approach
- System Architecture
- Kernel and Module Concepts
- Conclusion & Future Work
- Thesis Topics
New Requirements

- Requirements born by upcoming very (!) large scale projects and clinical trials
- **New** experimental data types generated by new high-throughput array techniques, e.g., tiling, exon, SNP arrays
- **New** chip producers, e.g., Illumina
- Integration of **third party** data management solutions, e.g., BC SNPMax
- Generic exchange of clinical data, e.g., by using XML
- Need for a **generic** approach to store all types of object groups
- User-right-management: rights per **user** on data and function
New Requirements cont.

- Flexible analysis integration (upload and execution of R analysis scripts)
- Move of (long running) analysis jobs to dedicated analysis servers
- Storage and return (on demand) of RData objects for every analysis
- Flexible analysis annotation (use of ontologies?)
- **Versioned integration** of web-data
- Web-GUI using Web 2.0 techniques
- Fat client on desktop (e.g., Rich Client)
GeWare 2.0

- Need for a redesign of GeWare (≤1.0) → GeWare 2.0
- **GeWare 2.0**: Mediator on warehouses
  - own warehouse (ext. DB schema of GeWare 1.0)
  - other sources of high-throughput data
GeWare 2.0

- Need for a redesign of GeWare (≤1.0) → GeWare 2.0
- GeWare 2.0: Mediator on warehouses
  - own warehouse (ext. DB schema of GeWare 1.0)
  - other sources of high-throughput data
- Generic source access layer
  - independence of source-specific query syntax
  - portability
**GeWare 2.0**

- **Need for a redesign of GeWare (≤1.0) → GeWare 2.0**
- **GeWare 2.0**: Mediator on warehouses
  - own warehouse (ext. DB schema of GeWare 1.0)
  - other sources of high-throughput data
- **Generic** source access layer
  - independence of source-specific query syntax
  - portability
- **Modularization**: Kernel and task-specific modules
  - system vs. plug-in modules
  - flexible module development (integrate a new or replace an existing module) by using a defined abstract module interface
Generic Source Access Layer

- Initialization by a set of data source connections
- Transparent data source access by kernel and modules
- Connection pooling for fast and parallel access
  - Problem: Which modules can store / retrieve data from which data source?
  - Solution: source-specific metadata

source-metadata(source id, module name, schema version)

- Mount (and unmount) data sources on system’s runtime
- Further problems:
  - Uniqueness of object identifiers
  - Selection of insert source
Uniqueness of Object Identifiers

- Problem: No unique object identifiers when multiple sources are used for same type of data
- Solution: Encoding and decoding of source-specific object identifiers

\[ f : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N} \]
\[ f(id_{src\, k-Obj}, k) = id_{sys-Obj} = id_{src\, k-Obj} \times c + k \]

\[ k \in \mathbb{N} : = \text{unique source identifier} \]
\[ c = 10^e, \, e \in \mathbb{N}, \, e > 0 (e.g., c = 1000) \]

\[ f_{src}^{-1}(id_{sys-Obj}) = k = id_{sys-Obj} \mod c \]
\[ f_{obj}^{-1}(id_{sys-Obj}, k) = id_{src\, k-Obj} = (id_{sys-Obj} - k)/c \]
Project Manager

- Project
  - typically, singular (planned) process for a specified time
  - in GeWare: Container for storing every type of data; project semantics keeps in user’s responsibility

- Problem: Selection of an insert source

- Solution:
  - Association of each project and module with a singular data source as selection for insert operation
  - read, update and delete for all other data sources

![Diagram of Project Management]

**Legend**
- Primary key
- Foreign key

<table>
<thead>
<tr>
<th>Project</th>
<th>1</th>
<th>Project Sources</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>project id</td>
<td>module name</td>
<td>data class</td>
<td>source id fk</td>
</tr>
<tr>
<td>name</td>
<td>description</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Experiment Manager**

- **Experiment** = Collection of
  - metadata (name, type, etc.) about high-throughput arrays, e.g., expression, exon, ... arrays
  - array-based experiment annotation
Experiment Manager

- **Experiment** = Collection of
  - metadata (name, type, etc.) about high-throughput arrays, e.g., expression, exon, ... arrays
  - array-based experiment annotation

- Experiment annotation on basis of *annotation templates*
  - set of hierarchically organized categories for which values can be captured
  - can be organized on pages
Experiment Manager

- **Experiment** = Collection of
  - metadata (name, type, etc.) about high-throughput arrays, e.g., expression, exon, ... arrays
  - array-based experiment annotation

- Experiment annotation on basis of **annotation templates**
  - set of hierarchically organized categories for which values can be captured
  - can be organized on pages

- Input values
  - manual input
  - selected **ontology** concepts (no flat vocabularies)
Module-specific Database Schema Portion

Legend:
- **Primary key**
- **Foreign key**

- **Experiment**
  - experiment id
  - name description

- **Chip Type**
  - type id
  - name description

- **Chip**
  - chip id
  - experiment id fk
  - type id fk
  - template id fk
  - data class
  - import date

- **Free Annotation**
  - category id fk
  - chip id fk
  - value

- **Rel Annotation**
  - category id fk
  - chip id fk
  - obj id fk

- **Logical Source**
  - lds id
  - obj type id fk
  - pds id fk

- **Physical Source**
  - pds id
  - name

- **LDS Object**
  - obj id
  - lds id fk
  - accession
  - name
  - description
  - is obsolete

- **LDS Structure**
  - lds structur id
  - lds id fk
  - parent obj id fk
  - child obj id fk
  - type id fk

- **Object Type**
  - obj type id
  - name

- **Category**
  - category id
  - page id
  - name

- **Relation Type**
  - type id
  - name
  - description

- **Template**
  - template id
  - name description

- **Page**
  - page id
  - template id fk
  - parent page id fk
  - name
**Feature Set Manager**

- Feature = **generic object** representing an object of interest, such as genes, proteins but also chips
  - analysis result
  - user-specified
**Feature Set Manager**

- **Feature** = *generic object* representing an object of interest, such as genes, proteins but also chips
  - analysis result
  - user-specified

- **Feature Set** = user-specified / pre-specified set of features

- **Capture**
  - semantics by using a meaningful object type and
  - data lineage given by a physical source
Feature Set Manager

- What’s GeWare
- Requirements
- GeWare 2.0
- Approach
- System
- Architecture
- Kernel and Module Concepts
- Conclusion & Future Work
- Thesis Topics

- Feature = **generic object** representing an object of interest, such as genes, proteins but also chips
  ✦ analysis result
  ✦ user-specified

- **Feature Set** = user-specified / pre-specified set of features

- Capture
  ✦ semantics by using a meaningful object type and
  ✦ data lineage given by a physical source

- Methods for
  ✦ retrieval and im-/export
  ✦ set manipulation, e.g., union, intersect, diff, majority
Module-specific Database Schema Portion

- Object Type
  - obj type id
  - name

- Logical Source
  - lds id
  - obj type id fk
  - pds id fk

- Physical Source
  - pds id
  - name

- Feature
  - feature id
  - lds id fk
  - accession

- Feature Set
  - set id
  - name

Legend:
- Primary key
- Foreign key

What's GeWare
Requirements
GeWare 2.0
Approach
System
Architecture
Kernel and Module Concepts
Conclusion & Future Work
Thesis Topics
**Annotation View Manager**

- **What’s GeWare**
- **Requirements**
- **GeWare 2.0 Approach**
- **System Architecture**
- **Kernel and Module Concepts**
- **Conclusion & Future Work**
- **Thesis Topics**

- **Annotation** = Description, explanation of specific object by using an attribute set
- **Annotation view** = defined set of attributes and corresponding values describing a primary (biological) object
Annotation View Manager

- Annotation = Description, explanation of specific object by using an attribute set
- Annotation view = defined set of attributes and corresponding values describing a primary (biological) object

<table>
<thead>
<tr>
<th>Probe Set ID</th>
<th>Gene Title</th>
<th>Gene Symbol</th>
<th>go biological process term</th>
<th>go molecular function term</th>
<th>go cellular component term</th>
<th>Pathway</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000_at</td>
<td>mitogen-activated protein kinase 3</td>
<td>MAPK3</td>
<td>protein amino acid phosphorylation</td>
<td>nucleotide binding</td>
<td></td>
<td>MAPK_Cascade, S1P_Signaling, TGF_Beta_Signaling_Pathway</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>protein amino acid phosphorylation cell cycle</td>
<td>protein kinase activity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>protein serine/threonine kinase activity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MAP kinase activity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MAP kinase activity</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>protein binding</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ATP binding</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Annotation View Manager

- Annotation = Description, explanation of specific object by using an attribute set
- Annotation view = defined set of attributes and corresponding values describing a primary (biological) object

Capture
- semantics by using a meaningful object type and
- data lineage given by a physical source
Module-specific Database Schema Portion

<table>
<thead>
<tr>
<th>Annotation View</th>
<th>View Attribute</th>
<th>Logical Source</th>
<th>Physical Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>view id</td>
<td>att id</td>
<td>lds id</td>
<td>pds id</td>
</tr>
<tr>
<td>name</td>
<td>view id fk</td>
<td>obj type id fk</td>
<td>name</td>
</tr>
<tr>
<td>version</td>
<td>lds id fk</td>
<td>name</td>
<td></td>
</tr>
<tr>
<td>date</td>
<td>name</td>
<td>is primary rank</td>
<td></td>
</tr>
</tbody>
</table>

**Primary Object**
- pobj id
- view id fk
- accession

**View Attribute**
- att id
- view id fk
- lds id fk
- name
- is primary rank
- rank
- value

**Logical Source**
- lds id
- obj type id fk
- pds id fk

**Physical Source**
- pds id
- name

**Legend**
- Primary key
- Foreign key
Web-Source Integration Manager

- Current idea: Use of BioFuice to integrate data from different source
- Implementation of integration workflows (iFuice scripts)
Web-Source Integration Manager

- Current idea: Use of BioFuice to integrate data from different source
- Implementation of integration workflows (iFuice scripts)
- Problems:
  - versioning of physical and logical sources
  - generation of annotation views
Web-Source Integration Manager

- Current idea: Use of BioFuice to integrate data from different source
- Implementation of integration workflows (iFuice scripts)
- Problems:
  - versioning of physical and logical sources
  - generation of annotation views
- Solution (First Approach)
  - versioning: physical source name = source name + version
  - view generation: new method genView() in BioFuice

\[ view = \text{genView}(M_1, \ldots, M_n, A_{s_i}) \]

where it exists a compositional path from mapping \( M_1 \) to \( M_n \) and set of attributes \( A_{s_i} \) from sources \( s_i \in (\text{domain}(M_j) \in \{M_1, \ldots, M_n\}) \) | \( \text{range}(M_j) \in \{M_1, \ldots, M_n\}) \)
Conclusion & Future Work

- Need for a redesign of GeWare based on new requirements
- GeWare 2.0: Mediator of warehouses
- Modularization: Kernel and task-specific modules
- Current state: Concept and implementation of system architecture, kernel and most modules
Conclusion & Future Work

- Need for a redesign of GeWare based on new requirements
- GeWare 2.0: Mediator of warehouses
- Modularization: Kernel and task-specific modules
- Current state: Concept and implementation of system architecture, kernel and most modules

Future work
- Need to implement: Experiment & Web-Source Integration Manager
- Need a detailed concept for analysis handling, monitoring, distribution (e.g., by using available Grid infrastructure), and annotation
Topics for Diploma/Master/Bachelor Thesis

- What’s GeWare
- Requirements
- GeWare 2.0 Approach
- System Architecture
- Kernel and Module Concepts
- Conclusion & Future Work

- Thesis Topics

- Web-GUI (re-)design
- Rich client application
- Application allowing to configure the clinical data exchange
- Analysis management and distribution
GeWare-Coders-Club

- What’s GeWare
- Requirements
- GeWare 2.0 Approach
- System Architecture
- Kernel and Module Concepts
- Conclusion & Future Work
- Thesis Topics
"Gehirnsturm am Freitagabend"

- What's GeWare
- Requirements
- GeWare 2.0
- Approach
- System Architecture
- Kernel and Module Concepts
- Conclusion & Future Work
- Thesis Topics