Goate: An infrastructure for new Web linking

Duncan Martin

Department of Computer Science and IT
University of Nottingham
Jubilee Campus
Nottingham
djm@cs.nott.ac.uk

Helen Ashman

Department of Computer Science and IT
University of Nottingham
Jubilee Campus
Nottingham
hla@cs.nott.ac.uk

Overview

This presentation covers:

- Low and high level linking
- Benefits of proxying
- Goate architecture
- Link presentation

HTML v XLink

HTML linking trails XLink in three ways:

- Links are uni-directional, not bi-directional
- Links are single, not multi-headed
- Only top-of-documents and fixed points can be referenced

Low level linking

We define a 'low level' linking language as having the following abilities:

- Move from one document to another
- Specify where in the destination to navigate to
- Create links visible in the destination document

Low level linking

< >

HTML meets this definition with the caveats:

- In-page destination point for the 2nd clause is pre-declared
- Writing <a href> links into the destination is allowed

Low level linking

Low level linking is important as it allows us to build more advanced linking behaviours.

This is analogous to the relationship between low and high level programming languages.

Low level programming languages are capable of all the same tasks as high level languages.

Emulating high level linking

< >

Higher-level behaviours can be emulated in HTML:

- Multi-headed links as a collection of single-headed links
- Bi-directional links as a pair of uni-directional links
- Flexible destination specification by placing in-page anchors in the destination document

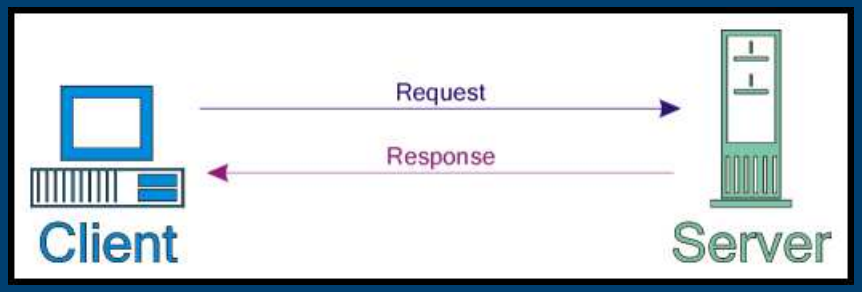
Introducing Goate

< >

Goate is a system that acts as a translator between high-level link specifications and HTML.

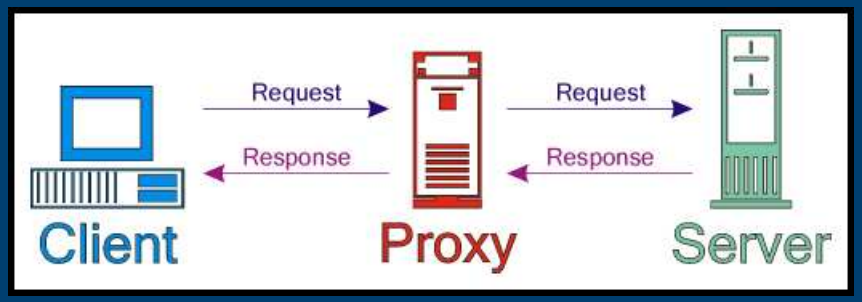
Goate operates as a HTTP proxy.

HTTP proxying



Without proxy

HTTP proxying



With proxy

Why a proxy?

Working a proxy has these advantages:

- No tie to browser or server platform
- Can intercept all communication
- Solves write-access problem

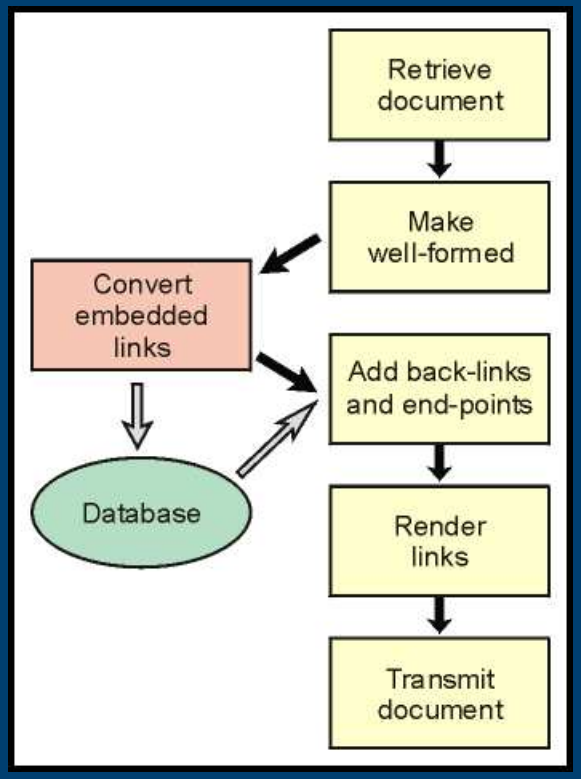
Goate architecture

Goate is written in ANSI C and runs under UNIX.

n copies are forked at startup.

Interfaces with SQL database (PostgreSQL).

Goate architecture



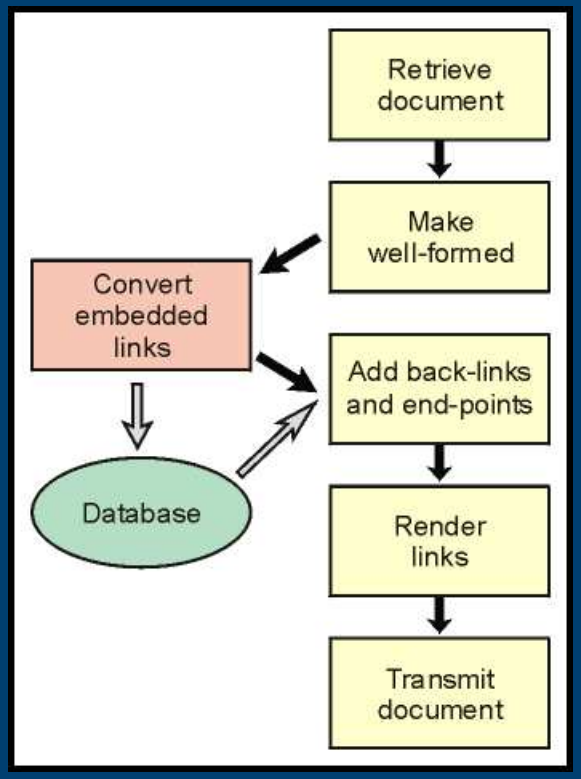
Language modules

The intepretation/semantic understanding of linking languages is done by language modules.

Modules linked into the system at run-time.

In the case of bi-directional links, and/or links with an in-page destination the module stores the end-point details in the database.

Goate architecture



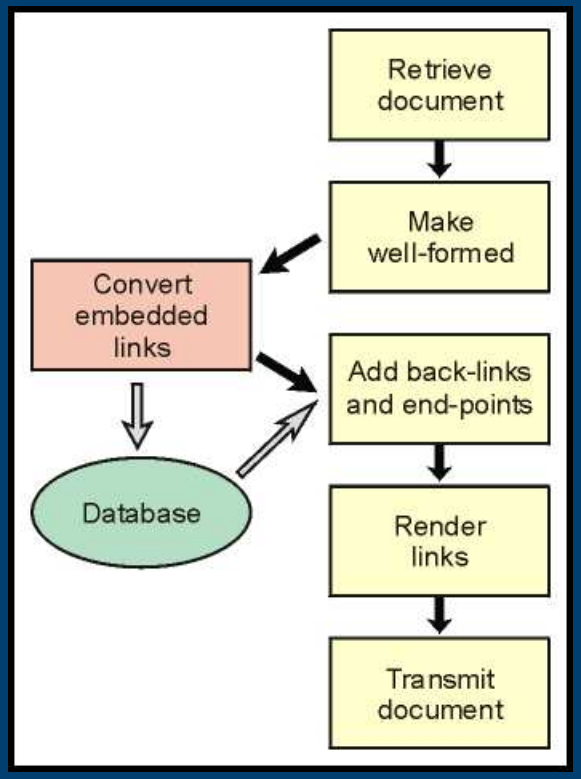
Add end-points and back links

The document we're processing may be the destination for other pages.

We need to add the in-page locators and the backwards part of bi-directional links.

A query is done on the link database looking for occurrences where 'destination page' is the current page.

Goate architecture



Link rendering

< >

Links in the document are converted into HTML.

Sections with overlapping links are rendered as multi-headed.

Precise code tailored to browser in use.

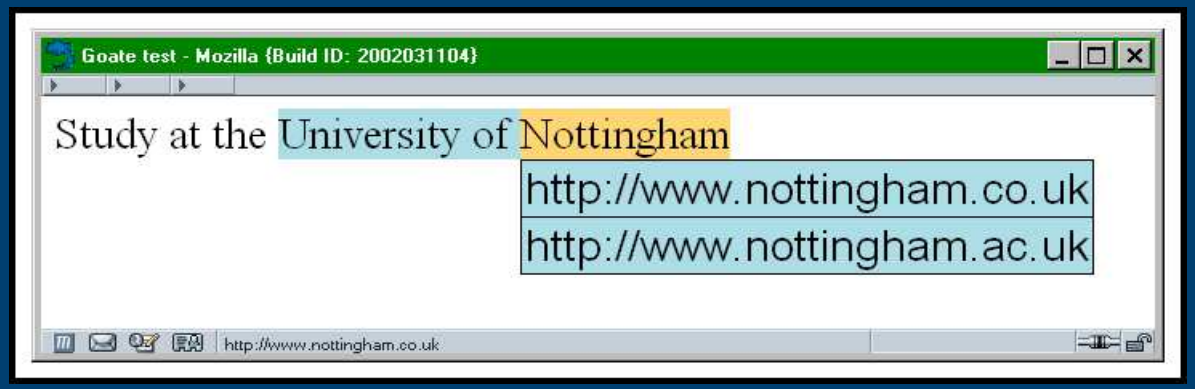
Link presentation

Links are rendered with background shading.

Different colours differentiate between:

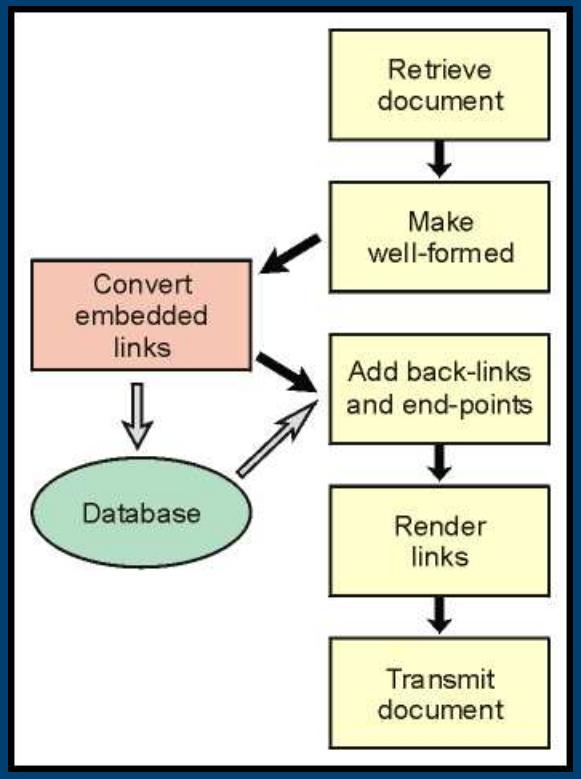
- Forwards and backwards links
- Single-headed and multi-headed links.

Link presentation



Mozilla screenshot

Goate architecture



Future work

Future work includes:

- Finish coding 'feature complete' version
- Publish a stable API for language modules, with full documentation
- Write interesting language modules

Any questions?

www.goate.org