Automated Localization Workflow – A Reference Model

How CMS and GMS collaborate in producing multilingual enterprise content.

From email to corporate medical policies, from license agreements to marketing brochures, from software to Web sites, e*nterprise content:*

- belongs to various domains: medicine, law, software, marketing, etc.
- is represented in various formats: Word, HTML, PDF, XML, etc.
- is stored in various repositories: files, databases, Content Management Systems (CMS), etc.
- is owned by various corporate entities: business units, departments, service centers, etc.
- can be located anywhere in the world, from desktops to enterprise servers

And all this content may, at any given time, need to be translated into any number of languages. It is no mystery that many companies find it difficult to control the translation/localization process, or even just to enforce common terminology across the enterprise. In fact, in many cases, it is difficult just to know where translation is happening and what the total cost is!

Content Management Systems (CMS) are systems designed to help author, store and deploy enterprise content. Globalization Management Systems (GMS) are systems designed to help manage and automate the localization process. Together, CMS and GMS provide a solution for the creation of multilingual enterprise content. This is illustrated by the high-level diagram in Figure 1:

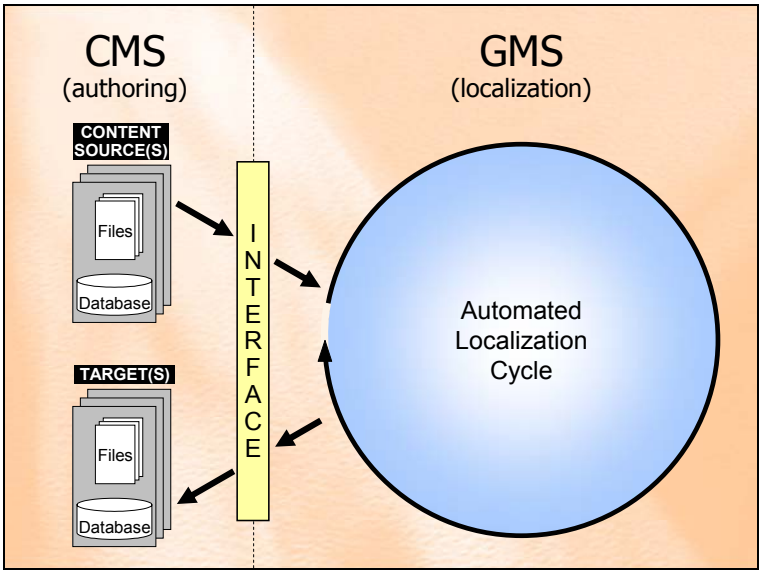

Figure 1 – High-level view of enterprise content localization

The CMS is involved primarily in the authoring of source content while the GMS is concerned mainly with managing the translation and localization of the content into any number of target languages. The resulting translated content is delivered back to the enterprise where it will possibly be managed by a CMS.

In some cases, the target and source content can all be stored in the same CMS at the same physical location. But remember that there are hundreds of commercial CMS and thousands of home-grown CMS solutions. In many cases the source and target content will be stored in several different CMS systems, possibly in several countries (for example when each country manages the local Web site, combining global templates, translated content from the head office and some local contributions).

Because enterprise content evolves continually, it is important that the localization process managed by the GMS be as automated as possible. GMSes use workflow technology to automatically detect new or modified source content, to automatically route the content to the localization vendors, to support the translation process with centralized linguistic assets and to automatically deliver the translated content back to the enterprise.

Both CMS and GMS are complex systems and therefore the way they collaborate, their interface, is non-trivial. Not only must they exchange content but they must also collaborate on the basis of rules and events. This will be explained further.

It is interesting to note that most GMS offer an architecture which allows several different sources of content to interact with a single localization management system. So while there may be several different CMSes involved, there is usually only one GMS. By centralizing all localization within a single system, the GMS offers the chance to put a bit of order in the growing body of content, to *reverse content entropy* as it were. A GMS can provide the enterprise with:

- centralized terminology management (useful for consistent global branding)
- centralized management of localization
- a single management interface for the localization process
- visibility of the localization process and its costs

The Automated Localization Workflow Reference Model

The reference model is presented in Figure 2; from the high-level view just presented, it drills down into a more detailed analysis focusing on the steps and components of the GMSes and their interface to the CMSes. Although the order of steps presented here is logical, different GMSes may use a different order, may group two or more steps into a single step or may not implement certain steps at all. The model is also independent of who actually performs the work. For example, if the customer is using a localization services firm, many steps will be performed by the vendor and others by the customer. When working with a small translation house, there will be a different sharing of responsibilities. The value of the model is that it covers the major issues involved in this process and provides a vocabulary to discuss these issues.

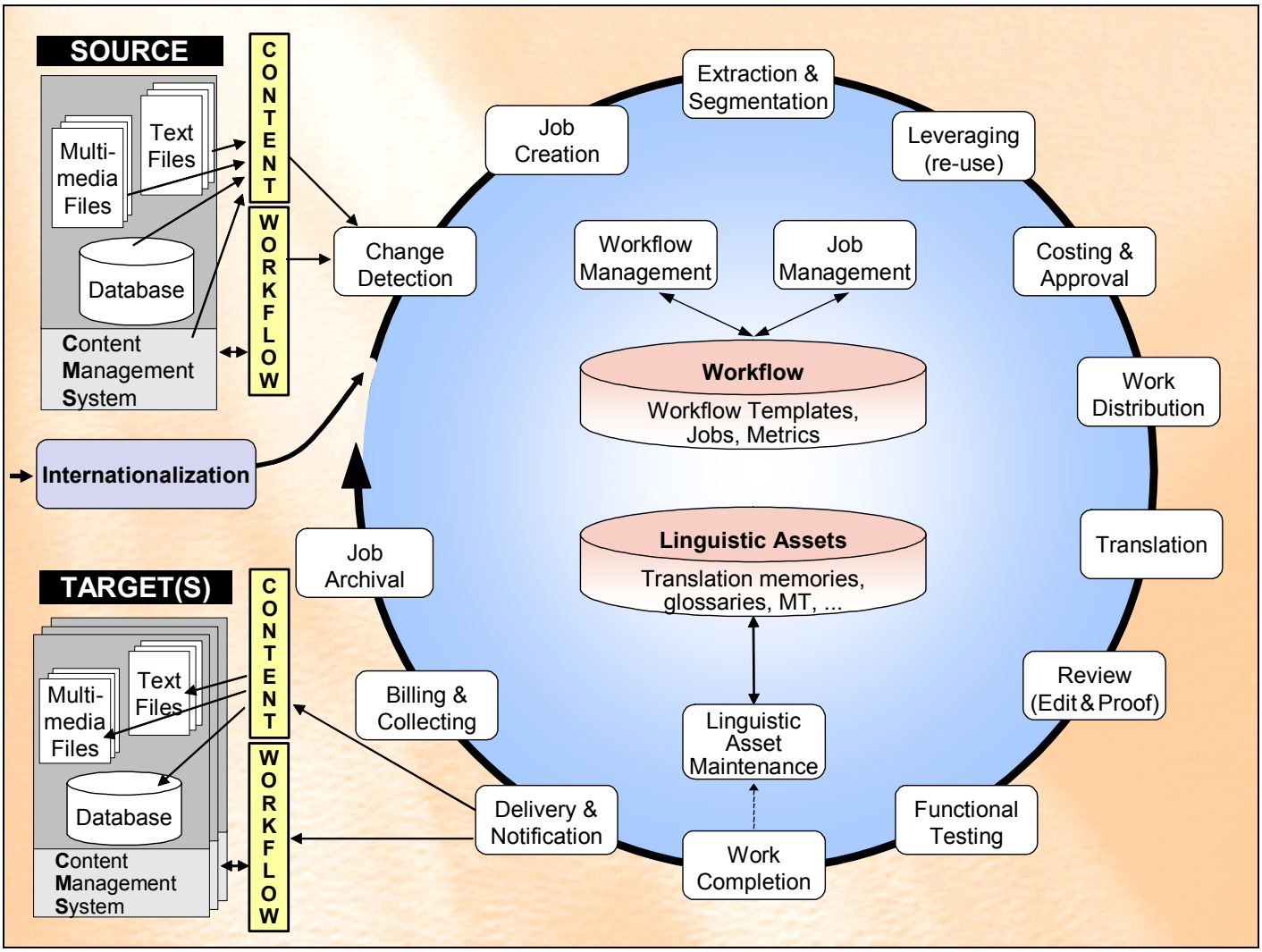

Figure 2: The Automated Localization Workflow Reference Model

SOURCE. As already mentioned, the source content comes in a variety of formats and can reside in many repositories. Here we show text files (HTML, XML, RTF, etc.) and multimedia files (Flash, Photoshop, etc.) and we could also mention templates (CSS, XSL, etc.) as interesting cases. The model also shows different repositories: files, databases and possibly a content management system. An important issue here is the *invasiveness* of the GMS: will the content have to move to another platform or database? Must tags be inserted in the files? Do the files have to be converted to a template based approach? Must the database schema be changed?

TARGET(S). The final objective is to deliver correct translated content in any number of languages to any number of repositories in any number of countries. The interesting question here is how source and target interact. In most organizations, for any given functional area, there will be one or more *primary content producers*. For example, in company X, most customer service information may be produced natively in German, while most legal documents may be produced in English. The content is then translated for use in other offices in other countries: the *consumers of translated content*. The question is how easily the consumers can create their own content to complement the translated content they receive. Furthermore, beyond a simple merge of local and translated content, there are more sophisticated and productive modes of collaboration possible: the consumers and producers can share terminology, document templates and even navigation structure (an approach pioneered by TRIDION). Note finally that, in the global enterprise, the consumers for some type of content may be primary producers for another kind of content.

Internationalization. Finally, the model reminds us that before the localization cycle can be truly functional, *all* content that needs to be localized should be properly prepared i.e. it should be *internationalized*. The CMSes involved must support data in a variety of languages and they may also offer localized user interfaces. Code, be it software programs or scripts in HTML pages, must be properly generalized and separated from the strings that need to be translated. Help files and documents should be rationalized and verified: there is no point in translating into many languages content that is wrong, redundant or simply a digression. Finally, for images containing text, providing just the final GIF or JPEG won't do; the original layered image files (Photoshop, for example) must be located and provided as the source content, otherwise the localization vendors will have to redevelop artwork just to change a few words (multiplying costs quite significantly).

Central Data Structures

All the localization steps shown in the model revolve around two major data structures: Workflow and Linguistic Assets. Workflow has to do with process automation, while Linguistic Assets help minimize translation work and ensure consistency. These data structures are usually stored in databases and sometimes in files. They can be located at the customer site, at the localization vendor's site, or even hosted by the GMS vendor (or any combination thereof). They store the basic objects of the system: workflow templates, jobs, translation memories, glossaries, etc. All these objects requires management and maintenance with the appropriate tool.

Workflow. In theory, the workflow engine is a general purpose tool that allows arbitrarily complex processes to be defined: any number of steps, any number of people involved anywhere in the world, any task be it manual or automated. In practice, there is a trade-off between complexity and ease-of-use: some systems have a fixed sequence of steps and the only configuration possible is to skip a step; other systems support general workflow templates with conditionals and have their own scripting language. It is important to remember that localization is the objective here, and that workflow is only a means to an end. Most GMSes allow deadlines to be defined, some also allow a budget; LTC, in particular, monitors budgeted vs. actuals and displays gross margin (for the localization vendor). Finally, *escalation* is an important feature i.e. if a task gets stuck, will it be re-routed to someone in charge, will an alert be sent to whoever is in charge? If not, then the system must be monitored very regularly to make sure everything is on track.

Workflow Management. This refers to the process of defining and maintaining the workflow templates that specify the steps that jobs in the system will execute. Some systems have wizards with only a few questions to answer, others require several pages of options to be set, others still have graphical interfaces that allow a process to be defined as a flowchart (and some also have a scripting language).

Job Management. As the system starts processing jobs, it must provide a management console to allow tracking and control over the jobs. It should be easy to see all jobs in the system and their status, in particular if they are stuck or behind schedule or if there are issues with the job. The control features should make it easy to get jobs unstuck, either by rerouting them or by canceling them. In general, customers do not want to manage localization; what they do want is better visibility into the process managed by the vendor. It's a good idea for these systems to provide two distinct interface for job management; one for the customer, with less detail and less control, one for the vendor with all details (including those, if any, he would prefer not releasing to the customer).

Linguistic Assets. These include one or more translation memories, glossaries and possibly automatic translation software. One could also include dictionaries, spell checkers and grammar checkers in this category (although these tools are generally presumed to be on the translator's desktop). Linguistic assets reduce the cost, increase the quality and increase the consistency of translation work. In the most general sense, linguistic assets store translation knowledge; if the system knows how to do something once, it may be able to do it again automatically (or at least make pertinent suggestions to the human translator). Some systems have one large centralized translation memory in Oracle, others will simply collect several project-based translation memories onto a file server. Some systems make it possible to download the pertinent subset (i.e. only those segments that match the text to be translated) of the translation memory in order to work offline effectively.

Linguistic Asset Maintenance. An often neglected point is that linguistic assets must also be maintained. The more work that is routed through the system, the more translation knowledge is accumulated, promoting more re-use. But as more and more data is accumulated, the system will also accumulate different translations for the same text segments. As translation knowledge grows, it becomes less precise and contains more "noise". Linguistic Asset Maintenance is required to avoid chaotic growth of translation knowledge and ensure that the captured data can be leveraged in a meaningful way.

Content & Workflow Interfaces

These are the logical components of the generalized interface. In fact, most systems will have a content-only interface for file systems and databases, and another content+workflow interface for CMSes and other such systems comprising workflow.

Content Interface. At a minimum the GMS must be able to retrieve and deliver content, be it stored in a file system or a CMS. Some systems exchange only files, using XML to represent database content. Others have more direct interfaces for databases. One notable exception is IDIOM: instead of copying the content over to the GMS, they have a sophisticated content interface which actually leaves the content wherever it is in the enterprise. This means that the filtering and segmentation of content is performed on the fly anytime the content is accessed by the leveraging engine, the translator, the reviewer, etc. This promotes a tighter collaboration between content authors and translators.

Workflow Interface. When a CMS and GMS work together, we basically have two workflow systems collaborating. The CMS will ensure that new source is authored, revised and possibly tested, and then this new content may be sent over to the GMS for translation. When the GMS sends the translated result back, the CMS may be in charge of the final deployment of the content. At a minimum, the Workflow Interface communicates events "source content needs translation" or "translation work done". Higher levels of integration are also possible; the CMS and GMS may also share a unified Job Management interface meaning the manager will be able to monitor translation projects from within the CMS user interface. A truly seamless interface will make the GMS appear as an extension of the CMS.

Automated Localization Cycle

The localization cycle is the sequence of steps any piece of content goes through in order to be localized.

Change Detection. The Change Detection module monitors the source content and is responsible for detecting changes and initiating action. Change monitoring may operate continuously or at regular intervals. File changes can be detected by file modification date, checksums, etc. Database changes are handled by adding to the tables special status fields or update triggers. Some systems offer a database connection wizard (IDIOM, for example), others consider database connectivity to be part of the system installation/customization phase.

Job Creation. The Job Creation module receives the changed (or new) content that needs to be translated along with its attributes: source and target languages, priority, deadline, preferred translators, etc. The content is grouped into meaningful *jobs* (also called *projects* or *orders*) for the workflow system; jobs should neither be too small (translators really are not interested in $5 jobs) nor too large. The jobs may be created on the basis of number of files, file types, target languages, priority, or time interval.

Extraction. As mentioned earlier, enterprise content comes in a variety of formats: Word, HTML, PDF, etc. Each format requires its own filter. The filters are responsible for extracting *(filtering out*) the actual text from its containing format, separating the source content into two parts: the text and the *skeleton*. The text portion may also contain basic text attributes such as bold and underline; the attributes may coded in a format-specific manner or normalized. SDL, for example, have a normalized approach, which allows a bold header in Word to match the same bold header in an HTML or PDF file. In order to preview the appearance of

the translated text, the skeleton must be merged with the translated text; The list of formats supported, the quality of support and the context (offline or online) in which the formats are supported varies from one system to another.

Segmentation. Once the text is extracted, it must be segmented into units of the appropriate size in order to be compared against the translated segments stored in the translation memories. The segmentation unit is often the sentence, but the exact set of rules for delimiting a segment varies from vendor to vendor. Sometimes information from the skeleton will be used to help define segments and segmentation will happen during extraction. Many tools use translation memories (in some form or other): TRADOS, Catalyst, SDL, STAR, Déjà Vu, GlobalSight, IDIOM, etc. and differences in segmentation algorithms mean that, when moving from one tool to another, a significant chunk of the translation memory can be, for all intents and purposes, lost (10% - 50% loss, depending on the particular combination of tools).

Leveraging. The Leveraging module tries to translate all source text segments using the Linguistic Assets. It may find exact and fuzzy matches in one or more translation memories or glossaries, it may also use machine translation. The end result is for each text segment, a list of matches and a classification of these matches into exact matches, fuzzy matches of various precision (90%, 80%, etc.), glossary matches, automatic translation "matches", etc. The match results will be provided to the translator either by storing them in a job-specific translation memory or glossary or by pre-translating the text segments.

Costing & Approval. All text segments classified by the leveraging step are word-counted. For each text segment, the system now has the word-count, the match class as well as source and target languages, priority, deadline, etc. All these factors can be combined to produce a cost for each item in the job. Note that most systems automate only word-based costing; costing based on page counts or costing for other tasks than pure translation has to be handled separately. Eventually, a quote is produced for the current job and routed to the customer for approval. Approval may be manual or automatic, i.e. the job gets automatically approved unless its size exceeds a certain threshold.

Work Distribution. Once a job has been approved, the work must be distributed to one or more translators in one or more countries. Most systems provide some form of vendor management database which will store a list of translators and agencies along with the language pairs they can handle. Some systems will also store the various tasks that a vendor can perform (translation, review, DTP preprocessing, bidi engineering, etc.). Most important to achieving a quality result is being able to store and manage a quality rating for each vendor as well as the domains (software, legal, medicine, etc.) that they can handle.

Translation. In this step, the translator actually translates the work received using the tools provided by the system or his own tools if the system can interface with them. This is likely the most important step as the main cost of localization is translation and the cost of translation is largely determined by how efficient an environment is provided to the translator. The translator may work *online* with a browser-based tool, or *offline* on his desktop PC. Browser-based tools are excellent for dealing with small changes, but for more substantial work, translators are usually more productive working on their desktops. Browser-based translation workbenches are not always feasible for remote translators working in countries where the Internet is slow or expensive. Note also that desktop tools connected to a GMS may have features missing or supported somewhat differently; notably fuzzy matches, glossaries, previewing, spell-checking, concordance searches, etc.

Review. The translation work is then routed for review (editing and proofing). The work is checked for translation accuracy and for overall document correctness. Most systems have no special review tools: the reviewer uses the same tool as the translator. Measuring language quality is notoriously difficult but is the only way process improvement can be achieved. Language quality metrics (SAE J2450 is one example) can be computed on statistically significant samples of the work before and after review. The system should allow for user-defined data to be stored in the database.

Functional Testing. Any form of content that is somehow rendered dynamically with programmable logic (scripts in HTML, content stored in a database, etc.) should then be tested before being published. Since customers often lack testers that speak the target languages, it is often most efficient for the localization vendor to provide the testing services. This usually requires a good collaboration between customer and vendor, to give the vendor access to the customer's staging site, for example, or to provide test plans, guidelines, even automated test scripts (that may themselves need to be localized). Some systems have features to track bugs and generate bugfix tasks in the workflow.

Work Completion. This step is actually quite simple: a reviewer or tester uploads a tested file or simply hits a "Done" button. The importance of this step is that it represents a milestone at which the translation work performed is deemed correct. It is at this time that the Linguistic Assets (translation memories, glossaries) are updated and that Linguistic Asset Maintenance is best performed.

Delivery & Notification. The final work must be delivered to the client enterprise through some form of content connector (CMS interface, database interface, etc.). Delivery should not interfere with local content development or site operation. Security is of paramount importance at this time, as this step not only adds new content, but may override existing enterprise content.

Billing & Collecting. Enterprise content grows and changes regularly and can generate a large number of translation jobs in the system. The billing and collecting for this translation work is either handled by some form of global agreement (time-based, volume-based or otherwise), or each job is billed separately. In the latter case, automation is essential as the system can process many jobs per day. Some systems will allow financial data to be dumped into XML files, possibly imported later into an Excel spreadsheet, while others have direct interfaces to systems such as Oracle Financials.

Job Archival. One of the advantages of a workflow system is that it makes it possible to easily accurate statistics (time spent, number of issues, % leverage, per content, per language, etc.) which can ultimately help improve the localization process. It should be possible to save and manipulate job data to this end.

Using the model

This generic model illustrates the major issues involved in automating localization workflow. It is useful in evaluating processes and tools like GMSes by providing a vendor-neutral terminology and conceptual reference. By drilling down one step further, the model has been used to generate an evaluation framework of nearly 1,000 questions which elaborate each step of the process and also include criteria to evaluate vendors: viability, R&D process, Quality Assurance, Customer Support, etc. This evaluation framework has been used to perform comprehensive evaluations of several vendors including GlobalSight, LTC, SDL, STAR, Trados (formerly Uniscape) and Tridion (as well as the former Convey and the late WorldPoint). Managers facing the multilingual challenge can use this model to better understand, challenge and evaluate the solutions proposed by the various vendors in order to select the appropriate process and technology for their organization.

Automating localization workflow is very complex and it may be that no system meets all your needs but, then, what other choice do you have?