DOI:10.1145/3132269

Article development led by acmqueue
queue.acm.org

Cardboard surrounds and protects stuff as it crosses boundaries.

BY PAT HELLAND

XML and JSON Are Like Cardboard

IN TODAY'S WORLD, cardboard is an ever-important part of life. Given the major investment of resources and money, you might question whether it's worth it. It turns out the efficiencies and savings from cardboard outstrip the costs to manufacture and later recycle it.

Semi-structured representations of data are not the cheapest format. There is typically a lot of extra stuff like angle brackets contained in it. JSON, XML, and other semi-structured representations allow for wonderful flexibility and dynamic interpretation. The efficiencies and savings gained from flexibility more than make up for the overhead.

Cardboard surrounds and protects stuff as it moves across boundaries. No one uses cardboard to move parts around within a factory. Instead, they use custom-designed containers that are specially purposed for the parts being produced. Cardboard is used to protect the stuff as it leaves the factory.

JSON and XML are used to protect data when it moves across trust boundaries. Semi-structured data wraps a single message or a single item in a key-value store in a way that allows for flexibility and extensibility. Inside an application, relational data is more tightly controlled and well formed. Evolving your relational data inside the trust and management boundary of an app is tractable.

SQL and relational data are easier and better for processing data *within a trust boundary*. XML and JSON are more flexible and dynamic as they capture the information and its metadata. This makes it easy and flexible to squirt data across trust boundaries.

Self-defining and self-identifying. Cardboard is usually self-describing. Your new TV has printing on the outside of the box telling you what's inside the box. As you move your old TV to your new home, you write "TV" on the outside of the moving box.

JSON and XML are usually self-describing. This can be done by referencing a schema or by examining the attributes expressed within the document/file itself.

Generic vs. custom. The last time we moved to a new home, I bought a bunch of boxes of varying sizes, tape, wrapping paper, and padding, and a bunch of marking pens. Like most other folks in the throes of packing and moving, we worked hard to describe the contents of every box we filled, but we occasionally messed up and omitted some items from the list as everything went into a box. Most things fit well into one of the standard boxes, although some of our household items involved really creative uses of cardboard, tape, and padding as we worked to protect our stuff.

Manufactured items frequently have custom-made cardboard protection. My wife loves the vacuum cleaners from one particular manufacturer. Indeed, the shape, form, and workings of the vacuums can be fun and surprising. To me, half the fun is disassembling the cardboard protection used inside the cardboard box. There are dozens of spe-

cial pieces of cardboard wedged into every nook and cranny of the vacuum. Man, that vacuum is well protected! I suspect they have a factory just to create the specialized pieces of cardboard. I also suspect the savings from avoiding damage are well worth it.

XML grew out of the document markup world. It descended from SGML (Standard Generalized Markup Language), which was originally intended to separate the text of a document from its formatting. XML is very strongly oriented around letting you "do your own thing" with the format.

Yet, on top of the flexible "do your own thing" approach, there are mechanisms to impose rigor and constraints on XML documents. XML Schema came into being in the early 2000s as a means of ensuring consistency for a set of messages. A document is validated if it conforms to an XML schema definition. In this way, some usages of XML are constrained to fit a particular shape and form.

One of the wonderful things about XML and JSON is their flexibility. In some applications, they support a tightly prescribed schema much like the cardboard protecting the vacuum cleaner. In other applications, they allow you to toss in all your family goods, including the kitchen sink. Sometimes, there is a tightly prescribed schema of required data while the sender can toss in extensions to its heart's content.

Crossing boundaries. In general, semi-structured data is used to cross boundaries in your computing environment. Documents containing human-readable stuff are kept on websites. REST calls are made across services that may or may not reside within the same company.

The loose coupling of semi-structured data allows the sending and receiving services to evolve separately with much lower friction. Changing tightly coupled stuff requires coordination that is just plain difficult.

Crossing boundaries with key-value stores. Frequently, semi-structured data in documents or files is stored in a file system or a key-value store. It is valuable to have readers and writers of these docs/files decoupled in their metadata. To have the shape and form of the data described in the contents of the docs and files makes it possible to evolve the various users with less friction than you would see if the metadata were strict and rigid. This is why we see the success of semi-structured representations for stored stuff.

It's not the size that counts. It turns out the weight and size of the cardboard are not that big of a deal. You have surely had the experience of receiving some small item such as a computer chip packaged in a box that weighs a *lot* more than the stuff being protected. It makes economic sense to protect the tiny thing well.

Large e-commerce sites ship tens of thousands of different things of different sizes. Still, they find it more efficient to use a relatively small number of box sizes. Consequently, it's common to open the box and find a tiny thing and a whole bunch of padding.

Similarly, you shouldn't be too worried about the bulkiness of your files and documents. The embedded metadata can take a lot of space. Lord knows, an XML file has a lot of angle brackets! Still, the value accrued from the features of semi-structured data is worth it. As long as the world doesn't run out of angle brackets, it will be all right.

Gotta take care of your stuff! In cardboard, the safety and care for stuff is the important reason for its existence. Similarly, in XML and JSON the safety and care of the data, both in transit and in storage, are why we bother.

Now, if only we could figure out efficient recycling for used angle brackets, we would be good to go …

IMAGES FROM SHUTTERSTOCK.COM

Related articles on queue.acm.org

The Power of Babble
Pat Helland
http://queue.acm.org/detail.cfm?id=3003188

Rules for Mobile Performance Optimization
Tammy Everts
http://queue.acm.org/detail.cfm?id=2510122

Schema.org: Evolution of Structured Data on the Web
R.V. Guha, Dan Brickley, and Steve Macbeth
http://queue.acm.org/detail.cfm?id=2857276

Pat Helland has been implementing transaction systems, databases, application platforms, distributed systems, fault-tolerant systems, and messaging systems since 1978. He currently works at Salesforce.