The underlying approach to evolving SData

Version: 1.0

Motivation

SData, now a few years old, is actively used by a number of teams across Sage. Since its release, there were a significant number of successfully delivered SData projects - just as there were some setbacks. Learning from past experiences, both positive and negative, we set to re-work the SData standard in the light of its fundamental role: to deliver the underlying protocol for Sage applications seeking web enablement and integration.

This paper outlines aspects considered by the SData working group to be fundamental for the version 2.0 of the standard.

SData 2.0 is a protocol toolkit

The path we originally followed with SData was to develop a fairly stringent definition frame, enabling Sage applications to produce reusable solutions. The primary goal was to provide the primitives for application integration, specifically the CRM-ERP integration. This orientation was reflected in many of the decisions taken by the standardizing body, leading to a result that was viable for application integration but too heavy for more casual scenarios.

There are a number of lessons learned from past experience, but probably the most fundamental is that there will always be a number of scenarios that are not be well served by rules aimed at solving a totally different use case. Recognizing this, the working group decided to take the approach of positioning SData as a protocol toolkit.

SData should consist of:

- a small, required set of essentials
- complemented by a wide variety of mechanisms and techniques that are solution orientated

The core is quickly described:

- compliance with HTTP rules and definitions
- adherence to one of the SData supported formats, ATOM or JSON

Most of the existing requirements are relaxed, allowing teams more freedom of decision in their implementations.

The mechanisms are solutions to general, day-to-day problems: how to do paging, how to invoke operations, etc. They represent a collection of ready-made solutions, potentially backed by community solutions available to SData implementers. There is no requirement to use a specific toolkit member, but existent solutions must be considered prior to designing new ones.

We believe that, by stressing the liberality of SData and allowing teams to select those aspects meaningful to them, we will see implementations that are light-weight, agile, quickly delivered and tailor-made to the business needs.

SData 2.0 maintains compatibility of existing solutions

SData 2.0 is no longer a 'green field' solution – it emerges in a setting where a substantial number of solutions are already present. Although SData is primarily forward looking, it should avoid forcing existing products to embark on a forced-change route without a well-founded reason.

The SData 2.0 features were carefully considered from the compatibility angle. It was the specific desire of the working group to deliver the message:

"Your SData 1.x compliant implementation is automatically compliant to SData 2.0"

To achieve this, we restricted ourselves to:

- relaxing existing rules by changing the compliance qualifier from MUST or SHOULD to MAY to increase the level of optionality
- making new features optional
- deprecating (not disallowing) outdated solution

SData provides technological choice

SData 2.0 is about choice. The most visible choice is the one between the xml-based ATOM format and the JSON format. This is a fundamental change to SData and an opportunity to confidently enter the arena of modern, web-based applications.

The JSON format allows SData to operate outside the strictures of the ATOM protocol - originally designed for news syndication. SData expands from the tools and formalisms present in the xml world, to the lighter and more dynamic world of JavaScript consumers. In this context, SData provides the structures necessary for efficient and flexible JSON-based processing. It should be noted however that not every JSON feature is immediately available in the xml format as well: metadata handling primitives and flexible replacements for XSD-based schemas are initially available on the JSON side only. The standard will close this gap in time.

On a wider scale, SData has only few requirements (HTTP, a few payload formatting aspects, some special syntax) but provides a variety of solutions to choose from: querying, paging, batching, reliable post, metadata handling and more. The latter is the wider - yet optional - offering of the SData 2.0 toolkit.

SData implementations rely on contracts

If SData is about choice, real-life implementations are about precision. From its beginnings, SData viewed contracts as the conceptual backbone of implementations; their role is to select the appropriate services and mechanisms whose combination delivers a practical business solution.

Under the impression that the concept of a contract requires no further clarification, the standard had omitted a clear definition – leaving room for misunderstanding.

SData continues to see contracts as the cornerstone of solutions. Version 2.0 describes the role and importance of contracts clearing the most common misconceptions that accompanied this concept up to date.

SData contracts can select from the variety of services and SData formalisms those features deemed necessary to achieve their goal. Contracts clearly state these choices and are free to impose more stringent rules do adopters than required by SData.

SData derives value from reuse

The advantage of the toolkit positioning is the flexibility it puts in the hands of delivery teams to select and implement those features deemed necessary for a solution. This promotes the creation of lean, purpose-directed solutions for the most varying use-cases.

As the number of SData-based implementations increase, value is derived from the technological foundation (HTTP, REST), and further, from the re-use of tried-and-tested mechanisms present in the standard.

The SData community is encouraged to exchange white papers, architectural approaches, extensions (when the standard is insufficient) and even components that emerge in the development process. To underpin the community aspect, the standard will be complemented with a knowledge repository allowing teams and individuals to find and benefit from the experiences of other community members.