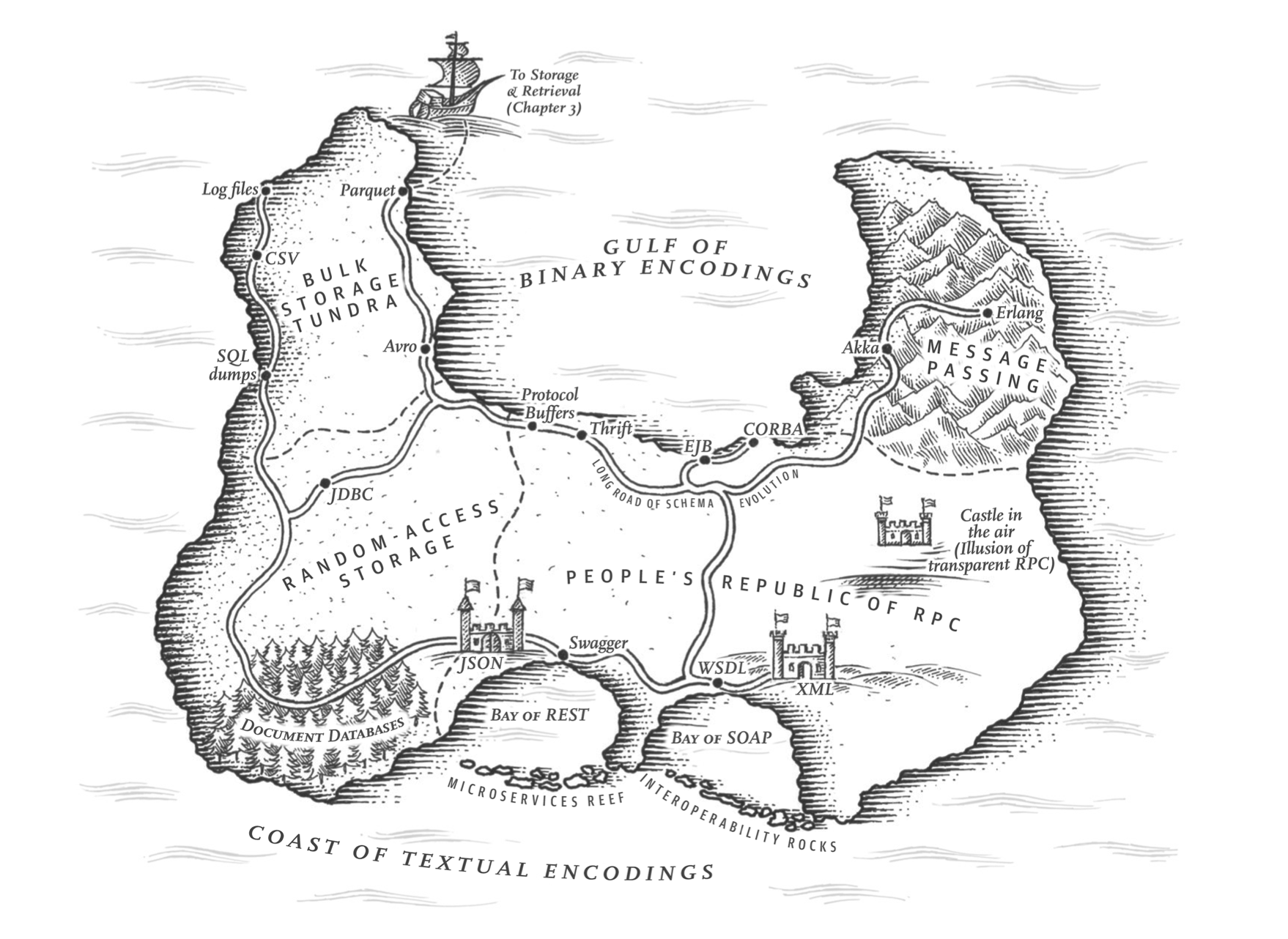
Chap 4: Encoding and Evolution.



* Backward compatity: new version writer, old reader
* Forward compatity: old writer, new reader.
* Avro Schema writer và schema reader không cần thiết phải giống nhau, chỉ cần compatible.
* Rule: chỉ remove hoặc add các field có giá trị default. Trong Avro thì để có thể là null thì field phải là dạng union.
* Avro có trick để thay đổi tên của 1 field, đặt alias cho new reader schema. Điều này dẫn đến là chỉ có thể backward compatible.
* Writer shema:

+ Avro lưu file lớn với cùng 1 schema, lúc này, chỉ cần để schema ở đầu file.

+ Lưu data theo version. Khi đọc 1 record thì đọc theo schema của version đó.

* Schema của Avro được tạo ra dynamically. Mỗi khi schema thay đổi thì chỉ cần generate lại schema.

1. Modes of Dataflows:

* Via database
* Via service calls
* Via asynchronous message passing.

1. Dataflow through database:

* Trở ngại lớn nhất là các process (chương trình) khác nhau có thể có các writer và reader khác version. Dữ liệu cũ phải immutable -> Avro, Parquet có thể phù hợp.

1. Dataflow through Service: REST, RPC

* SOAP Webservice vs REST API: SOAP thì dùng XML, REST thì có nhiều loại. SOAP thì có cấu trúc phức tạp hơn, khó construct bằng tay được, phụ thuộc vào tools. REST thì đơn gỉan hơn.

1. Remote procedure call:

* Try to make a request to a remote network service look the same as calling a function in programming language. Điểm khác biệt ở chổ là:

+ Local func thì predictable, success hoặc fail phụ thuộc vào đối số truyền vào. Remote func thì còn phụ thuộc nhiều yếu tố nữa.

+ Local func thì có thể trả về result, throw exception, hoặc không return (vì crash hoặc nhảy vào loop vô tận). Remote func thì đôi khi không trả về result chỉ vì timeout.

+ Khi retry 1 request, đôi khi việc trùng lặp sẽ xảy ra. Như vậy phải có giải pháp dedupication trong protocol. Local func thì không gặp điều này.

+ Thời gian chạy 1 local func thì khá tương đồng nhau. Thời gian chạy network thì thường chậm hơn, và biến động (variable). Có thể do network congestion.

+ Parameter của local func thì có thể truyền theo kiểu pointer. Tuy nhiên với remote func thì tất cả đều phải encode. Điều này gặp trở ngại lớn với những object cực lớn.

+ Client và service có thể được viết bằng các ngôn ngữ khác nhau, nên RPC phải translate được datatypes giữa 2 ngôn ngữ.

REST tỏ ra nhỉnh hơn hẳn trong việc phát triển public APIs. RPC thì focus vào việc request giữa các services trong cùng 1 organization, trong cùng 1 data center.

1. Message-Passing Dataflow:

* Sử dụng Message broker thì có các ưu điểm hơn RPC:

+ Hoạt động như 1 buffer, nếu như recipient (bên nhận) bị quá tải hoặc ngưng hoạt động. Do đó tăng độ tin câyj.

+ Có khả năng gửi lại message nếu process bị crashed.

+ Hạn chế việc sender phải biết IP và port của recipient.

+ Cho phép gửi 1 message cho nhiều recipient.

+ Decouple sender với recipient. Sender chỉ cần gửi message, không quan tâm đến ai sẽ consumes nó.

* Tuy vậy cơ chế message thì không thể làm cơ chế send-response. Sender chỉ gửi và không expect phải nhận được respónse lại.

1. Message brokers:

* RabbitMQ, ActiveMQ, HornetQ, NATS, Apache Kafka là các nền tảng message brokers open-source.

1. Distributed actor frameworks:

* Akka
* Orleans
* Erlang OTP.

