The Business Transaction Protocol

The business transaction protocol was designed to enable fast and efficient deployment of applications which are tailored to managing information exchange between internal or external business entities involved in a transaction.
To better the protocol, certain definitions need to be put forward.

Business Service: A business service consists of a set of activities performed at a cost and following a specified procedure. For the purpose of our application development, every business service should have appropriate request(s) that demands an appropriate response(s).

> ***"Every business services consist of activities performed at a cost and requiring feedback (proof of transaction)"*** *- Aniefiok Friday*

Transaction: A transaction will be defined as any exchange of value between the participating entities. The value exchange may be done for monetary or non monetary purposes.

Business Object: This forms the base data model of the protocol. All information are encapsulated and transported in a business object instance. The business object forms the base class for all data model used in the BTP and it is implemented as a dictionary like container that is read-only and only modified via an **update(**parameters)** method. Though a business object may be extended to have methods fitting for target application it is best to keep them as read-only dictionaries. This pratice, will enable separation of concerns and use of framework functions for data manipulation.

BUSINESS TRANSACTION PROTOCOL - BRIEF INTRODUCTION

The aim of BTP is to simplify application development by 2 key factors:
1. Enable exchange of data by using only one type of data object or model.
2. Improve communication between various aspect of application layers by transforming or acting on the same data model.
By keeping all data in preferably flat, parse, and mutable data model information can be readily made available. Development team are then allowed to focus primarily on functions that generates, consumes, altars this data model.

All data in the Business Transaction Protocol can be categorised into 3 groups:

i. Request Object:
   This object is used to encapsulate any system user request or sub-system request. User in this context can be a human agent, third party interface requesting information, or a scheduled task requesting data.

ii. Action or Performance: This too is a subclass of the *BusinessObject*. The Action class models data after it has been processed by a method or function. Example, a user request may be processed by function X (this may be specified in the request through named flags or by designer) which will not simply return a value but a Performance object that holds/references information of the request and outcome of processing the request. Performance object can be passed to other functions or series of functions before finally achieving a response for the request.

iii. Response Object: The response object constitute two major parts:

- The message which serves as feedback to the user
- The document which constitute a comprehensive data object of system response to a given request. This data object can be parsed as electronic document, email, SMS, of printable PDF.

BTP follows a development cycle which is:
Request -> Performance (s) -> Response.

By keeping this convention, all business services being executed by an application can be broken into 3 phases namely: **requesting, acting, responding**. Software developers can then be tasked with providing code facet for delivering on the 3 aspects of the protocol. Furthermore, the various application sections can be segmented as shown in table below:

APPLICATION LAYER	BUSINESS OBJECT	FUNCTION
Web front-end	RequestObject e.g HttpRequest	django view
Database (queries)	RequestObject e.g SQL query	model query methods
Database (records)	Performance e.g model instance	model manager methods, model methods, model properties/fields
Web page (refresh)	ResponseObject e.g returned HttpResponse, JSON response, PDF file download	view resource, file stream, output stream, error, log, database update

BENEFITS

The following benefits are inherent in the BTP approach.

i. Decompose complex business applications into 3 primary sections
ii. Simplify user activity with application by having predetermined requests and their corresponding performance functions
iii. Tailor responses to user requests which can be centrally managed or dynamically generated from a template
iv. Ease communication between diverse application layer by using same base object for conveying information
v. Attempt to separate data from behaviour keeping applications as collection of functions instead of a monolithic package