who exactly needs a protocol as a service?

In a market made of *decentralized* services, everything is a protocol. i made this observation to BillG in a ThinkWeek paper entitled protocol-oriented programming (which got some good attention in 2002). A similar observation was made by VMWare's RabbitMQ team nearly 5 years later. Another 5 years later the blockchain community has come to the same conclusion. Witness the language of the Ethereum project.

In reality, protocols have been important to computing services since the dawn of telephony. What's different now is that with the blockchain we have a mechanism for building a *censorship-resistant, global public* compute infrastructure that can't be taken down by private or public sector attacks. Thus, decentralized services that don't require the same kind of trust in 3rd parties become economically feasible.

which existing protocols don't work and why? (need examples)

There are few examples of successful distributed and decentralized protocols.
There are no examples of widely adopted decentralized social networks.
There are no examples of widely adopted decentralized music sharing networks.
There are no examples of widely adopted decentralized freelancer or on-demand services.
There are no examples of widely adopted decentralized dating networks.

There are many examples of failed or struggling projects to create these. The incumbent versions of these services (Facebook, Twitter, Spotify, Freelancer.com, OKCupid, etc) are all centralized and require a high degree of trust from the user base. The number of stories of this trust being abused (by those organizations or the NSA) is staggering as well sobering. However, what galvanizes the user base to seek market alternatives (see the initial market responses to Diaspora, Ello, etc) is that they are unable to participate in the economics in a way that they deem fair. In other words, the incumbents not only abuse trust and are subject to devastating security attacks that put millions of users' sensitive data at risk. They also take the lion's share of the profits.

There is essentially one example of a widely adopted decentralized value transfer application, i.e. Bitcoin and its lookalikes. Although bitcoin itself is successful, it is struggling to evolve and scale.

why do peer apps require new protocols?

Distributed and decentralized apps have always required good protocols. There is an opportunity to make protocols even easier to build, more reliable, scalable. In addition, more complex and valuable software can be built when there emerges a reliable collection of interoperable and pluggable protocols on the market. Think about the protocols to revolutionize the agriculture industry in developing countries, off of cash and onto a smart contract system -- many protocols involved.

LivelyGig (marketplace for on-demand talent) is planning protocols for: Agent, Introduction, Invitation, Verification, Reputation, Contract, Bitcoin Escrow, Post Query, Venue Channel, Yenta Matchmaker, Expiration, and others.

doesn't the W3C take care of this stuff?

The W3C is largely a standards body. Understandably, they move much more slowly and deliberately. With the advent of a censorship-resistant, global public compute infrastructure enabling all of the application opportunities listed above this space is going to be volatile for the next many years with protocols being proposed and evaluated by the market at a much more rapid pace. This is going to be 1999 on steroids.

how can you tell for certain that I need to build a custom protocol?

will it make my apps faster? Yes, for certain cross-globe patterns of interaction.
... more reliable? Yes, especially when deployed and managed in a decentralized manner
... more secure? Yes. data is highly obscured and can also be encrypted by the user

why can't i use one of the existing decentralized app development platforms in the market?

Which one would that be? There are no successful general-purpose platforms; many are trying. Ethereum is biggest projects although it has yet to prove itself. Greg is very actively engaged with the Ethereum and they are adopting the principles of applied pi calculus. In addition, Ethereum's consensus protocol will directly influence Special-K as well.

how much time will I save by developing using your platform versus another (realistically)?

It's really a matter of you couldn't do this sort of thing before the advent of this technology. Before this platform the developer just couldn't build these kinds of applications.

if "the math is the code", does some of the validation work actually go away?

Yes! It becomes part of the type declarations and type checking. In other words, it's automated, and woven into developer standard practice. In particular, SLMC, Pro Verif, and other model-checking and theorem-proving mechanisms are part of a larger trend to provide large scale protocol verification.

since most systems that are already built are centralized, and since decentralized systems that are being built need to connect to them, what advantage does Special K have for enabling hybrid systems?

Agreed that some protocols that are architecturally decentralized will have adapters to centralized data and systems. Another approach will be for oracles to call into the APIs on decentralized systems. There is nothing special in that other than it allows value emanating from centralized systems to have greater reach, and an evolution path toward a more complete decentralized system.

Additionally, SpecialK's specific architectural choices and components have been designed with just this in mind. The architecture is modular and components (which facilitate service level integration) can be swapped. Additionally, there is support for integrating at all levels of the stack.

- It's built on a best-of-breed JVM language (Scala), and there are language bindings for the APIs.
- Its messaging system uses the AMQP standard and best-of-breed implementation (RabbitMQ), and the APIs are exposed at that layer as well.
- It provides an HTTP bridge.